Teilnehmerrechensysteme und stochastische Entscheidungsmodelle

Fritz Lehmann

1972

TEILNEHMERRECHENSYSTEME UND STOCHASTISCHE ENTSCHEIDUNGSMODELLE

Inaugural-Dissertation
zur Erlangung der Doktorwürde
der Fakultät für Mathematik der
Ludwig-Maximilians-Universität
München

Vorgelegt von Fritz Lehmann im Dezember 1971

UNIV.
LEIDEN
BIBL.

1. Berichterstatter: Prof. Dr. G. Seegmüller

2. Berichterstatter: Prof. Dr. H. Richter

Tag der mündlichen Prüfung: 14. Juni 1972

DISSERTATIONS– UND FOTODRUCK FRANK oHG
8 München 2, Gabelsbergerstraße 15, Tel. 2 8 0 9 0 9 0

Inhaltsverzeichnis

		Seite
0.	Zusammenfassung	1
1.	Betriebsweisen und Organisation von Rechenanlagen	4
1.1	Betriebsweisen aus der Sicht der Benutzer	4
1.2	Maschineninterne Organisation	8
1.3	Probleme bei Teilnehmerrechensystemen	13
1.4	Das Organisationsprogramm	21
1.5	Zuteilungsalgorithmen einiger realisierter Teilnehmerrechensysteme	26
1.6	Theoretische Arbeiten über Zuteilungsalgorithmen	33
2.	Grundlegende Begriffe aus der Theorie der stochastischen Entscheidungsmodelle	36
2.1	Vorbemerkungen	36
2.2	Definition stochastischer Entscheidungsmodelle	39
2.3	Einige bekannte Hilfssätze	41
2.4	Optimalitätsbegriffe	44
3.	Ein allgemeines mathematisches Modell	50
3.1	Heuristik und mathematische Formulierung des Modells	50
3.2	Bewertung von Politiken	59

3.3 Optimierungsmodelle aus der Literatur 62

3.4 Bemerkungen zur praktischen Anwendbarkeit des Modells 65

4. Betriebsmodell und stochstische Entscheidungsmodelle 69

4.1 Reduktion des Betriebsmodells auf ein stochastisches Entscheidungsmodell 69

4.2 Existenzsätze und die Optimalitätsgleichung 73

4.3 Spezielle Entscheidungsmodelle und Politiken 77

5. Ein konkretes Modell 88

5.1 Allgemeine Formulierung 88

5.2 Einige Ergebnisse 103

5.3 Endliche Zustandsmengen 105

5.4 Zwei-Benutzer-Modell 109

6. Literaturverzeichnis 137

0. ZUSAMMENFASSUNG

In einem Teilnehmerrechensystem können von mehreren Benutzerstationen aus unabhängig Aufgaben abgewickelt werden. Ein Organisationsprogramm übernimmt dabei die Koordination der gleichzeitig anstehenden Aufgaben und stellt die zu ihrer Bearbeitung benötigten Betriebsmittel gemäß einer bestimmten Betriebspolitik bereit. Eine allgemeine Theorie zur Behandlung der dabei auftretenden organisatorischen Probleme ist zur Zeit nicht bekannt. In der vorliegenden Arbeit wird aber ein mathematisches Modell für den Betriebsablauf in einem Teilnehmerrechensystem konstruiert, das verschiedene Präzisierungen der Optimalität und einige Aussagen zum Problem der optimalen Festlegung von Betriebspolitiken gestattet.

Im ersten Kapitel werden die Organisation und verschiedene Betriebsweisen von Rechenanlagen beschrieben. Dabei wird das Unterbrechungskonzept in neueren Datenverarbeitungsanlagen als unentbehrlich für die Tätigkeit des Organisationsprogramms herausgestellt. Man kann die Aufgabe des Organisationsprogramms weitgehend in der Verarbeitung der stochastisch eintreffenden Unterbrechungssignale sehen. Diese Beobachtung bildet die Grundlage für das zu konstruierende Modell.

Im zweiten Kapitel werden stochastische Entscheidungsmodelle im Sinne von Hinderer (1967) eingeführt, die in enger Beziehung zu unserem Modell stehen.

Im dritten Kapitel wird dann ein mathematisches Modell für die Arbeitsweise des Organisationsprogramms konstruiert. Eine Betriebspolitik für das Organisationsprogramm ist dabei durch eine Vorschrift festgelegt, die zu jedem Unterbrechungssignal die randomisierte Wahl einer Aktion aus einer Menge von jeweils zulässigen Aktionen bestimmt, wobei das Auswahlverfahren von der gesamten Vorgeschichte abhängen kann. Das stochastische Verhalten des Systems wird durch Übergangswahrscheinlichkeiten beschrieben. Zur Bewertung von Betriebspolitiken werden die Kosten benutzt, die beim Betriebsablauf durch den mit den Aktionen verbundenen Verwaltungsaufwand, unerwünschte Verzögerungen von Benutzerprogrammen und andere beim Betrieb auftretende Ereignisse entstehen. Die meisten theoretischen Arbeiten zum Problem der Betriebsmittelvergabe benutzen Warteschlangenmodelle oder Simulationsstudien, die jedoch jeweils nur einzelne, von vornherein festgelegte Betriebspolitiken behandeln können, wogegen das hier konstruierte Modell die Gesamtheit der sinnvollen Betriebspolitiken enthält. Die verhältnismäßig wenigen Untersuchungen, die das Problem der Optimierung behandeln, benutzen speziellere Modelle. So weit wir sehen, können sie alle mit unseren Hilfsmitteln erfaßt werden.

Im vierten Kapitel wird das konstruierte Modell auf ein stochastisches Entscheidungsmodell zurückgeführt, wodurch bekannte Sätze der Theorie dieser Modelle zur Gewinnung von Aussagen über optimale Betriebspolitiken herangezogen wer-

den können. In der Literatur ist dieser Zusammenhang anscheinend noch nicht bemerkt worden. Die Theorie der stochastischen Entscheidungsmodelle zeigt insbesondere, daß man sich in vielen praktisch wichtigen Fällen auf deterministische Politiken beschränken kann und daß diese häufig auch nur einen Teil der Vorgeschichte berücksichtigen müssen. Aussagen dieser Art kommen im Zusammenhang mit Rechensystemen in der Literatur anscheinend ebenfalls noch nicht vor. In speziellen Modellen sind auch Konstruktionsverfahren zur expliziten Bestimmung optimaler Betriebspolitiken bekannt.

Im fünften Kapitel wird eines von ihnen benutzt, um in einem einfachen konkreten Modell für ein Teilnehmerrechensystem, bei verschiedenen Annahmen über die Kosten- und Wahrscheinlichkeitsgesetze optimale Betriebspolitiken zu studieren. Es erweisen sich je nach den benutzten Voraussetzungen verschiedene Prioritäts- und Time-Sharing-Verfahren als optimal. Time-Sharing-Verfahren werden bisher in realen Teilnehmerrechensystemen benutzt, ohne daß man sich über ihre Optimalitätseigenschaften voll im klaren ist.

Die Konstruktionsverfahren sind ziemlich aufwendig, was ihre Anwendung bei komplexen Systemen stark behindert. Andererseits laufen sie aber nicht während des Betriebes des Systems, sondern vorher bei der Festlegung der Betriebspolitik ab. Das Organisationsprogramm hat nur noch die Systemzustände und die Signale auszuwerten und die jeweils optimale Aktion einzuleiten.

1. BETRIEBSWEISEN UND ORGANISATION VON RECHENANLAGEN

1.1 Betriebsweisen aus der Sicht der Benutzer

Die programmgesteuerten digitalen Rechenanlagen haben sich aus relativ einfachen, leicht zu überschauenden Geräten zu komplizierten Datenverarbeitungsanlagen entwickelt, die sich aus einer Vielzahl weitgehend unabhängiger Komponenten zusammensetzen. Der Wunsch, diese Anlagen wirtschaftlich möglichst gut auszunutzen, wirft erhebliche organisatorische Probleme auf. Versuche zur Lösung dieser Probleme spiegeln sich in den verschiedenen Betriebsweisen von Rechenanlagen wider (vgl. z.B. Corbató 1963, Fano, Corbató 1966, Paul, Urich 1966, Spies 1967, Wilkes 1970).
Die ursprüngliche Betriebsart einer Rechenanlage ist der sogenannte "open shop"-Betrieb. Hierbei begibt sich der Benutzer mit seinem Programm an die Rechenanlage und nimmt alle manuellen Operationen, die zur Ausführung seines Programms an den Geräten notwendig sind, selbst vor. Er hat dabei den großen Vorteil, die Bearbeitung seines Problems in fast allen Einzelheiten unmittelbar mitzuerleben. Auftretende Fehler und Unregelmäßigkeiten kann er meist sehr schnell erkennen und durch geeignete Maßnahmen korrigieren. Voraussetzung für diese Betriebsart ist, daß die Benutzer technisch genügend ausgebildet sind, um vernünftig an der Anlage arbeiten zu können. Das ist nur bei einem kleinen Benutzerkreis möglich. Vom wirtschaftlichen Standpunkt aus

ist der "open shop"-Betrieb vertretbar, solange der Bedarf an Rechenzeit nicht allzu groß ist und die Verluste an Rechenzeit, die dadurch entstehen, daß die Anlage auf Aktionen des Benutzers warten muß, nicht sehr ins Gewicht fallen. Das trifft nur für langsame Rechner zu.

Durch die Fortschritte der Technik sind jedoch immer schnellere Rechner verfügbar geworden, die Hunderttausende von Operationen in der Sekunde ausführen können. Zugleich ist der Bedarf an Rechenzeit gewaltig gestiegen, so daß es nicht mehr tragbar ist, den schnellen Rechner auf die langsamen Reaktionen des menschlichen Benutzers warten zu lassen. In der gleichen Zeit, die der Mensch für die einfachste Überlegung benötigt, kann der Rechner eventuell mehrere kurze Programme anderer Benutzer ausführen.

Aus diesen Gründen ist die heute meist übliche Betriebsweise schneller Rechenanlagen der sogenannte "closed shop"-Betrieb. Dabei wird dem Benutzer der direkte Zugang zur Maschine verwehrt. Er muß vielmehr sein fertiges Programm in einer standardisierten Eingabeform beim Rechenzentrum abgeben und warten, bis er nach einiger Zeit, die Stunden oder bei großem Andrang gar Tage betragen kann, die Resultate oder Fehlermeldungen seines Programmlaufs zurückerhält. Die Benutzerprogramme werden vom Personal des Rechenzentrums eines nach dem anderen in den Rechner eingegeben. Diese Art der Bearbeitung wird häufig als Stapelverarbeitung oder "batch processing" bezeichnet.

Die Auslastung der Rechenanlagen wurde durch den Übergang zum closed shop-Betrieb erheblich verbessert. Für den Benutzer jedoch bedeutet er eine erhebliche Behinderung. Zwar wird er ein wenig dadurch entschädigt, daß vom Rechenzentrum eine Fülle von Dienstleistungsprogrammen angeboten wird, wodurch es zum Beispiel möglich ist, ein Programm statt in der für den Menschen doch ziemlich unhandlichen Maschinensprache in einer dem Problem entsprechenden, gewohnteren Sprache zu formulieren. Auch stehen eine Reihe besonderer Fehleranalyseprogramme zur Verfügung. Aber trotzdem benötigt ein Benutzer oft Wochen und Monate, bis er sein Problem gelöst hat, weil er das Programm, das ihm dabei helfen soll, sehr oft ablaufen lassen muß, um es auf seine Richtigkeit zu prüfen. Dabei muß er oft viele Stunden warten, bis er einen kleinen Fehler entdecken und beseitigen kann, den er, könnte er den Programmablauf unmittelbar miterleben, sofort behoben hätte.

Es entsteht daher der Wunsch, auch an einem großen Rechner dem Benutzer die Möglichkeit zu geben, in der gleichen Weise zu arbeiten, wie er es an einer Anlage täte, die nur für ihn da ist. Diesem Zweck sollen Vielfachzugriffsysteme dienen. Bei diesen ist eine Vielzahl von Bedienungsstationen an einen großen zentralen Rechner angeschlossen. Man unterscheidet weiter Teilhabersysteme, bei denen die Benutzer von den verschiedenen Bedienungsplätzen nur Zugriff zu bestimmten, fertigen Programmen haben, wie es zum Beispiel in Platzbuchungs- und Auskunftsystemen der Fall ist, und Teil-

nehmersysteme, bei denen die Benutzer ihre Programme selbst entwickeln. Bei letzteren hat jeder Benutzer von seiner Bedienungsstation aus unmittelbaren Zugriff zu fast dem gesamten Dienstleistungsspektrum eines großen Rechners. Dieser reagiert auf die Erteilung eines Auftrags in so kurzer Zeit, daß jeder Benutzer die Vorstellung gewinnen kann, die Anlage arbeite nur für ihn. So wird es dem Benutzer möglich, alle Phasen seines Programmlaufs mitzuerleben und zu fast jedem Zeitpunkt verbessernd einzugreifen, so wie es seiner natürlichen Arbeitsweise entspricht. Besonders in der Testphase eines Programms ist eine solche Möglichkeit von unschätzbarem Wert. Ein ausgetestetes Programm kann man dann auch im closed shop-Betrieb bearbeiten lassen, da bei ihm ein unmittelbares Eingreifen in der Regel nicht mehr nötig ist.

Schatzoff, Tsao und Wiig 1967, Sackman, Erikson und Grant 1968 und Gold 1969 haben experimentelle Untersuchungen zum Vergleich des Stapelbetriebs und des Teilnehmerrechenbetriebs angestellt. Es benötigten zum Beispiel bei Sackman, Erikson und Grant Benutzer zum Austesten ihrer Programme zur Lösung ausgewählter Probleme im Teilnehmerbetrieb je nach Aufgabe nur 2/3 bis 1/6 der menschlichen Arbeitszeit, die im Stapelbetrieb benötigt wurde.

Bisher wurde das Bemühen um Wirtschaftlichkeit des Rechnereinsatzes nur aus der Sicht der Benutzer geschildert. Es ist aber klar, daß der Übergang zum closed shop-Betrieb oder gar zu einem Vielfachzugriffsbetrieb auch intern hohe Anforderungen an die Organisation stellt.

1.2 Maschineninterne Organisation

Eine heutige Datenverarbeitungsanlage besteht aus einer Anzahl verschiedenartiger Komponenten, deren Zusammenspiel auf manche Weise organisiert werden kann.
In der Frühzeit der Rechenanlagen waren die Hauptbestandteile einer Anlage der aus Rechen- und Leitwerk bestehende Rechnerkern, der Zentralspeicher und eine bescheidene Ausstattung an Ein- und Ausgabegeräten. Die Zusammenarbeit dieser Teile war denkbar einfach: Die Befehle des Rechenprogramms wurden streng sequentiell abgearbeitet. War ein Ein- oder Ausgabebefehl auszuführen, so mußte der Rechnerkern auf die Beendigung dieses Vorgangs warten, ehe er im Programm fortfahren konnte. Die Hardware ließ nämlich nur jeweils eine aktive Komponente auf den Speicher zugreifen.

Diese Betriebsart weist nun intern den gleichen Mangel auf, der extern zur Einführung des closed shop-Betriebs geführt hat. Hinsichtlich der Arbeitsgeschwindigkeit besteht zwischen dem schnellen Rechnerkern und den viel langsameren mechanischen Ein- und Ausgabegeräten eine starke Diskrepanz, die dazu führt, daß der schnelle Rechnerkern einen beträchtlichen Teil der Betriebszeit auf die langsamen Anlagenteile warten muß. Hinzu kommt, daß die Zahl der mit dem Rechnerkern verbundenen externen Geräte im Laufe der Zeit stark angewachsen ist. Bei einer streng sequentiellen Abarbeitung der Programme kann aber immer höchstens eines dieser Geräte tätig sein, so daß der übrige Teil der Anlage zwar betriebs-

bereit aber ungenutzt ist.

Um nun die Leistungsfähigkeit des schnellen Rechnerkerns besser auszuschöpfen und die Gesamtanlage gleichmäßiger auszulasten, möchte man diese Wartezeiten des Rechnerkerns für andere Programme ausnutzen. Die Techniken, die diesem Ziel dienen, werden mit Mehrprogramm-Betrieb (multiprogramming) bezeichnet.

Dazu wird die Bearbeitung der Programme dezentralisiert. Die Ausführung der eigentlichen Ein/Ausgabeoperationen wird sogenannten Kanalwerken (vgl. z.B. Hassitt 1967, Flores 1969, Schecher 1970) übertragen, die ein eigenes Leitwerk besitzen und daher weitgehend unabhängig vom Rechnerkern und unabhängig von einander arbeiten können.
Da nun verschiedene tätige Anlagenkomponenten zum Hauptspeicher zugreifen, ist ein besonderes Vorrangwerk erforderlich, das die Zugriffswünsche von den verschiedenen Kanalwerken und vom Rechnerkern koordiniert.
Das Zusammenspiel der Zentraleinheit mit den Kanalwerken macht wesentlich von der Möglichkeit Gebrauch, daß die Kanalwerke den Rechnerkern bei der Bearbeitung eines Programms unterbrechen können, um auf Ereignisse bei der Ein- und Ausgabe aufmerksam zu machen. Etwas vereinfacht lassen sich diese Vorgänge wie folgt beschreiben.

Im Hauptspeicher der Anlage, der natürlich eine hinreichende Kapazität aufweisen muß, hält man eine Anzahl von Benutzerprogrammen zur Ausführung bereit. Der Rechnerkern arbeitet zunächst an einem dieser Programme. Wird von diesem nun

die Ausführung eines Eingabe- oder Ausgabebefehls verlangt, so gibt die Zentraleinheit diesen Befehl mit der notwendigen Information über das gewünschte Gerät und die zu transportierenden Datenmengen an das zuständige Kanalwerk weiter und fährt in der Bearbeitung des Programms fort. Jetzt arbeiten also Kanalwerk und Zentraleinheit an demselben Programm. Das geht so lange, bis die Ausführung des Programms an einer Stelle anlangt, an der der Ein/Ausgabevorgang beendet sein muß, damit eine Weiterarbeit sinnvoll ist. Ergibt nun eine Anfrage des Programms, daß der Ein/Ausgabevorgang noch nicht beendet ist, so unterbricht der Rechner die Arbeit an diesem Programm, wobei er dessen aktuellen Zustand für eine spätere Fortsetzung konserviert. Er wendet sich nun einem anderen der im Speicher bereitstehenden Programme zu. Dieses Programm kann seinerseits wieder Ein/Ausgabevorgänge veranlassen, die in der gleichen Weise behandelt werden.

Die Aufgabe, das Nebeneinander der verschiedenen Vorgänge zu koordinieren, wird von einem Organisationsprogramm übernommen. Die Zusammenarbeit des Rechnerkerns mit den Kanalwerken geht nur über das Organisationsprogramm. Nur dieses nämlich ist über die jeweils aktuelle Verteilung der Komponenten genau informiert. Erhält es vom rechnenden Programm einen Ein/Ausgabebefehl, so prüft es mit Hilfe von Listen, die es führt, nach, ob ein Gerät der gewünschten Art verfügbar ist. Ist dieses der Fall, so kann der entsprechende Kanal den Vorgang ausführen, wobei das Organisationsprogramm in seinen Listen die entsprechende Belegung registriert.

Ist kein Gerät der geforderten Art frei, so merkt das Organisationsprogramm den Ein/Ausgabebefehl für einen späteren Zeitpunkt vor.
Beendet nun ein Ein/Ausgabegerät seine Arbeit (oder sind während seiner Tätigkeit Fehler aufgetreten, die das Kanalwerk nicht beheben konnte), so unterbricht das Kanalwerk durch ein Signal (Eingriff) das gerade laufende Programm in der Zentraleinheit und aktiviert das Organisationsprogramm. Dieses untersucht die vom Kanalwerk mitgelieferte Information und stellt fest, welches Programm den eben beendeten Ein/Ausgabevorgang veranlaßt hatte. Wartet dieses Programm bereits auf die Vollendung dieses Vorgangs, so kann es jetzt fortgesetzt werden. Für das gerade noch vom Rechner bearbeitete Programm dagegen werden die aktuellen Registerinhalte abgespeichert, so daß es zu einem späteren Zeitpunkt an der Unterbrechungsstelle weitergeführt werden kann. Auf diese Weise können also mehrere Programme zeitlich ineinander verzahnt bearbeitet werden. Sind diese hinsichtlich ihrer Forderungen an die Rechenanlage genügend unterschiedlich, so kann eine gleichmäßige Ausnutzung aller Anlagenteile erreicht werden.

Eine noch größere Bedeutung als im soeben geschilderten Fall gewinnt das Organisationsprogramm bei der Realisierung von Vielfachzugriffsystemen. Es hat ja jetzt außer für eine geordnete Abwicklung der Ein/Ausgabebefehle und der Verwaltung der Anlagenkomponenten für eine schnelle Reaktion gegenüber den Benutzern an den Konsolen zu sorgen. Um jedem

Benutzer an einer Konsole die Illusion zu vermitteln, der Rechner arbeite nur für ihn, kann man auf zweierlei Weisen vorgehen. Entweder ist der Rechner so schnell und die Aufträge sind so kurz, daß sie alle innerhalb von Sekunden erledigt sind. Dieser Fall liegt zum Beispiel bei Platzbuchungs- und Auskunftsystemen (vgl. z.B. Dubner u. Abate 1970) vor, oder wenn das System nur zur Ausbildung von Studenten mittels kleiner Programme benutzt wird (vgl. z.B. Nickel 1968). In einem solchen Fall ist eine besondere Organisation der Rechenzeitverteilung nicht notwendig. Muß man dagegen mit Programmen rechnen, die einen erheblichen Rechenzeitaufwand erfordern, so muß man die Aufträge im Time-Sharing-Betrieb bearbeiten, wenn man den Konsolbenutzern rasche Reaktionen des Systems bieten will.

Im Time-Sharing-Betrieb wird an dem Programm eines Benutzers jeweils nur ein kleines Zeitquantum lang gearbeitet, bevor es zugunsten eines anderen Programms unterbrochen wird. So wird innerhalb einer ganz kurzen Zeit an jedem Programm gearbeitet, so daß Benutzer mit kurzen Anfragen sehr rasch bedient werden.

Das Organisationsprogramm hat nun neben der Verwaltung der zur Anlage gehörenden Geräte auch noch die Verteilung der Rechenzeitquanten vorzunehmen und dafür zu sorgen, daß die verschiedenen, Benutzerwünschen entsprechenden Programme, die laufend eintreffen, sich nicht gegenseitig zerstören und daß jedes nach seinen Bedürfnissen so schnell wie möglich erledigt wird.

1.3 Probleme bei Teilnehmerrechensystemen

Nachdem die vorhergehenden Abschnitte eine Motivation und eine sehr grobe Skizze der Realisierung verschiedener Betriebsweisen von Rechenanlagen gebracht haben, soll nun ein Überblick über einige Probleme bei Teilnehmerrechensystemen folgen.

Die Vielfachzugriffsysteme unterscheiden sich von herkömmlichen Organisationsformen von Rechensystemen, wie bereits mehrfach betont wurde, dadurch, daß viele Benutzer über abgesetzte Bedienungsstationen und Fernleitungen mit dem Rechner "gleichzeitig" verkehren, die Anlagenkomponenten sequentiell oder zeitlich verschränkt gemeinsam benutzen und sich nicht merklich gegenseitig behindern. Bei den Teilhabersystemen (vgl. z.B. Nolle 1970) hat der Benutzer Zugriff zu speziellen in der Anlage gespeicherten Programmen. Die Reaktionszeit des Systems ist nicht besonders kritisch. Ein Beispiel bildet ein Platzbuchungssystem, bei dem Anfragen, Vorbestellungen und Rückgaben von Karten von bestimmten, fertigen Programmen durchgeführt werden. Dem Kunden am Schalter, für den diese Aktionen durchgeführt werden, kommt es in der Regel auf eine halbe Minute nicht an.

Bei Teilnehmerrechensystemen dagegen können die Benutzer ihre Programme selbst entwickeln. Im Gegensatz zu den Teilhabersystemen ist hier der Rechenzeitbedarf der Programme nicht mehr im voraus bekannt. Zudem muß man mit fehlerhaften Programmen rechnen, während bei Teilhabersystemen die Program-

me im allgemeinen ausgetestet sind. Da die menschlichen Benutzer mit dem Rechner einen Dialog führen wollen, sind die als erträglich empfundenen Reaktionszeiten des Systems auf einfache Anfragen kürzer als bei Teilhabersystemen. Aus der Fülle interessanter Probleme bei Teilnehmerrechensystemen interessieren uns hier besonders die Probleme der Betriebsmittelverteilung. Diese lassen sich aber im Grunde von den übrigen Problemen nicht völlig trennen. Einige solcher Wechselwirkungen sollen daher hier angedeutet werden.

Ein dialogfähiges Teilnehmerrechensystem muß, damit es von seinen Benutzern über die Konsolen bedient werden kann, über eine Kommandosprache verfügen, in der der Benutzer seine Wünsche formulieren kann. Außerdem sollte es Übersetzer für einige höhere Programmiersprachen besitzen, die im Idealfall dialogfähige Objektprogramme erzeugen können sollten. Es ist klar, daß der Kreis der Benutzer, der von einem Teilnehmersystem angezogen wird, sehr stark davon abhängt, welche Hilfsmittel und welche Bequemlichkeit ihm hier geboten werden. Die Zusammensetzung der Benutzergemeinschaft bestimmt aber wiederum die Art der geforderten Dienste, die wiederum den Entwurf des Organisationsprogramms beeinflußt. Ein wichtiger Gesichtspunkt bei Teilnehmerrechensystemen ist auch die Sicherheit des Systems, die besonders bei der Technik des Mehrprogrammbetriebs dadurch gefährdet ist, daß durch Fehler eines Benutzers Programme oder Daten eines anderen Benutzers oder gar Betriebsprogramme der Anlage zer-

stört werden könnten. Dieser Gefahr muß man durch einen wirksamen Speicherschutz begegnen, der bei neueren Maschinen bereits weitgehend von der Hardware übernommen wird. Mit dem Speicherschutzproblem nähern wir uns der internen Betriebsmittelverwaltung, die der normale Benutzer gewöhnlich gar nicht direkt bemerkt.

Wie schon in 1.2 bemerkt, erfordert die Notwendigkeit, für alle Konsolbenutzer eine erträgliche Reaktionszeit zu erzielen, im allgemeinen eine Unterbrechung laufender Programme zugunsten eines wartenden Programms. Man wird nun nicht alle konkurrierenden Programme im Arbeitsspeicher halten können. Ein Teil von ihnen wird vielmehr auf einem Hintergrundspeicher abgelegt sein. Geht nun die Zentraleinheit von einem Programm auf ein anderes über, so ist häufig damit das Hinausschreiben des unterbrochenen Programms und das Hereinholen eines neuen Programms oder Teile eines solchen vom Hintergrundspeicher in den Arbeitsspeicher verbunden. Ein derartiger Austauschvorgang erfordert eine Verwaltungstätigkeit des Organisationsprogramms, das hierfür eine gewisse Zeit den Rechnerkern belegt und dadurch Benutzerprogramme verzögert. Weitere Rechentotzeiten können dadurch entstehen, daß der Rechnerkern auf die Vollendung eines Austauschvorgangs warten muß. Schließlich ist auch das den Austausch durchführende Kanalwerk belegt und steht für diese Zeit nicht für Datentransporte zur Verfügung, die von Benutzerprogrammen gewünscht werden. Insofern bedeutet also ein Programmaustausch eine starke Belastung des Systems. Beim

Entwurf des Zuteilungsalgorithmus steht man daher einerseits vor der Notwendigkeit, die Benutzerprogramme in möglichst kurzen zeitlichen Abständen zu unterbrechen und so eine kurze Reaktionszeit des Systems zu gewährleisten, auf der anderen Seite treten aber mit häufigeren Unterbrechungen auch die Austauschvorgänge häufiger auf. Bei zu kurzen Unterbrechungsintervallen besteht die Gefahr, daß das System sich im wesentlichen mit dem Austausch von Programmen zwischen Arbeits- und Hintergrundspeicher beschäftigt. Hier liegt also für die Zeitzuteilung ein Optimierungsproblem vor, bei dem diese beiden Gesichtspunkte gegeneinander abgewogen werden müssen.

Bisher haben wir nur den Übertragungskanal zum Hintergrundspeicher als Engpaß dargestellt. Aber wenn man in Teilnehmerrechensystemen auch Mehrprogramm-Betrieb zur besseren Ausnutzung der verschiedenen Anlagenteile anwenden will, bildet die Verteilung des Arbeitsspeichers ein weiteres Problem, denn man muß dann ja mehrere Programme zugleich im Arbeitsspeicher halten. Da die Lage im Arbeitsspeicher, die einem Benutzerprogramm zugewiesen wird, von der jeweiligen Betriebssituation abhängt, kann sie dem Benutzer nicht bekannt sein. Die Umwandlung der von ihm in seinem Programm benutzten Adressen, den sogenannten virtuellen Adressen, in die Bezeichnung der jeweils entsprechenden Kernspeicherzelle, die reale Adresse, wurde in früheren Anlagen beim Laden des Programms in den Kernspeicher durchgeführt. Da dieser Vorgang in Teilnehmerrechensystemen wegen der Austauschvorgänge

zum Hintergrundspeicher sehr häufig ist und die Umwandlung wegen der möglicherweise veränderten Lage des Programms bei jedem Laden neu erfolgen müßte, wäre dieses Verfahren zeitlich zu aufwendig. Man fixiert deshalb die realen Adressen erst zum Zeitpunkt der Ausführung der Befehle. Die Programme sind dann relativ adressiert bezogen auf ihren Beginn. Sie werden dann bei manchen Anlagentypen in einen zusammenhängenden Speicherbereich geladen, wobei die Anfangsadresse dieses Bereichs von einem besonderen Register, dem Basisregister aufgenommen wird. Vor der Ausführung eines Befehls, der eine Speicherzelle benutzt, wird auf die programmrelative Adresse der jeweilige Inhalt des Basisregisters aufaddiert, so daß immer der richtige Operand angesprochen wird. Beim Wechsel von einem Programm zum nächsten muß dann nur dieses Basisregister umgeladen werden.

Die Forderung nach zusammenhängenden Arbeitsspeicherbereichen führt bei diesem Verfahren auf die Schwierigkeit, daß die Speicherbereiche, die durch ausscheidende Programme freigegeben werden, verstreut im Speicher liegen, so daß nach einer Weile der Speicher zerstückelt erscheint und ein neues Programm nicht geladen werden kann, weil nirgends ein hinreichend großer zusammenhängender Speicherbereich zu finden ist, obwohl insgesamt vielleicht noch wesentlich mehr Platz vorhanden ist. Von Zeit zu Zeit muß man daher die im Speicher vorhandenen Programme zusammenschieben, um neue zusammenhängende Bereiche zu schaffen. Bei der Verschiebung muß

aber sichergestellt werden, daß durch die vorzunehmenden Veränderungen keine anderen Vorgänge, wie z.B. Ein/Ausgabeaktivitäten in den betreffenden Speicherbereichen, gestört werden. Das alles bringt aber eine neue Belastung des Systems mit sich.

Durch das Konzept der Seitenadressierung (vgl. z.B. Fotheringham 1961, Kilburn, Payne, Howarth 1962, Dennis 1965, Arden, Galler, O'Brien, Westervelt 1966, Randell, Kuehner 1968) versucht man diese Schwierigkeiten zu reduzieren. Dabei wird der verfügbare Speicherplatz in gleich lange Einheiten aufgeteilt. Im Raum der virtuellen Adressen heißen die Einheiten Seiten, im physikalischen Speicher Kacheln. Den Seiten eines Programms werden nun Kacheln zugewiesen, die nun nicht mehr zusammenhängend liegen müssen. Die virtuellen Adressen sind nun in eine Seitenadresse und eine Zeilenadresse innerhalb der Seite zerlegt. Die Zeilenadresse bleibt die relative Adresse innerhalb der zugewiesenen Kachel. Die Zuordnung der Kacheln im Speicher zu den Seiten der Programme wird mittels Tabellen vorgenommen. Die einer virtuellen Adresse entsprechende Hauptspeicherzelle wird dann auf folgende Weise bestimmt. Zunächst entnimmt man einer besonderen, dem Programm zugeordneten Liste die Stelle der Seiten-Kachel-Tabelle, die sich auf diese Seite des Programms bezieht. Die Seiten-Kachel-Tabelle liefert dann die Anfangsadresse der gesuchten Kachel, die man nun nur noch um die Zeilennummer zu erhöhen braucht, um die richtige Speicherzelle zu erreichen.

Dieses Verfahren vereinfacht die Verwaltung des Hauptspeichers natürlich gewaltig. Man braucht auch die Programme nicht mehr vollständig zu laden, sondern man kann sich jeweils auf die gerade benötigten Seiten beschränken und bei Bedarf erst weitere zuladen. Auf diese Weise kann der virtuelle Adressenraum den tatsächlich vorhandenen Speicherplatz erheblich übersteigen. Auch der Speicherschutz, Kontrollen zur Vermeidung von unnötigen Transporten unveränderter Seiten u.a. lassen sich in Verbindung mit dem Seitenkonzept in einem gewissen Umfang bequemer realisieren. Der Zuteilungsalgorithmus muß nun aber mit einer Reihe weiterer Probleme zurechtkommen. Wenn das Zeitquantum eines Programms abgelaufen ist, steht er vor der Frage, ob er jetzt eines der Programme starten soll, von denen bereits Seiten im Speicher stehen, oder ob ein neues vom Hintergrundspeicher geladen werden soll. Eventuell lassen sich solche Dinge parallel machen, indem Programm A dem Rechnerkern übergeben wird und gleichzeitig Seiten von Programm B gegen Seiten von Programm C aus dem Hintergrundspeicher ausgewechselt werden. Wenn die Seiten eines neuen Programms oder neue Seiten eines laufenden Programms geladen werden müssen, hat der Zuteilungsalgorithmus zu entscheiden, welche der im Arbeitsspeicher vorhandenen Seiten hinausgeschafft werden müssen. Im ungünstigsten Fall könnte es geschehen, daß im nächsten Augenblick gerade die soeben entfernte Seite gebraucht wird. Zum Seitenaustauschproblem sind einige theoretische Arbeiten erschienen (z.B. Belady 1966, Smith 1967, Denning 1968, Aho, Denning und Ullman 1971).

In einem System, das mehrere um die vorhandenen Betriebsmittel konkurrierende Aufträge im Mehrprogrammbetrieb bearbeiten soll, stößt man auch auf das Problem des zirkulären Wartens ("deadlock"-Problem; vgl.z.B. Havender 1968, Habermann 1969). Es könnte zum Beispiel geschehen, daß ein Auftrag ein Betriebsmittel A belegt und nun dazu ein Betriebsmittel B benötigt, um sinnvoll weiterbearbeitet zu werden. Das Betriebsmittel B ist aber gerade von einem zweiten Auftrag belegt, der aber nun zusätzlich das Betriebsmittel A anfordert. Nimmt man an, daß Aufträge ihre Betriebsmittel erst dann wieder freigeben, wenn sie sie nicht mehr benötigen, so haben wir eine Situation, in der die beiden Aufträge sich gegenseitig vollständig blockieren. Die Gefahr, daß derartige zirkuläre Wartezustände entstehen, wächst natürlich mit der Zahl der gleichzeitig vorhandenen Aufträge. Der Zuteilungsalgorithmus muß daher so angelegt sein, daß zirkuläre Wartezustände nicht auftreten können.

Wir wollen an dieser Stelle nicht weiter auf die Details dieser Probleme eingehen. Man findet sie zum Teil sehr ausführlich bei Wilkes 1970 behandelt. Hier sollte nur deutlich gemacht werden, daß eine Fülle von Gesichtspunkten die Wahl des Zuteilungsalgorithmus beeinflußt.

1.4 Das Organisationsprogramm

Die Aufgaben und die Arbeitsweise des Organisationsprogramms sind bereits grob skizziert worden. Doch um eine spätere Heuristik für ein mathematisches Modell zu erleichtern, erscheint es zweckmäßig, etwas detaillierter darauf einzugehen.

Von den verdrahteten Funktionseinheiten, der Hardware, wird die Arbeit des Organisationsprogramms durch die Unterbrechungstechnik möglich gemacht. Dabei kann ein im übrigen unabhängig vom Rechnerkern arbeitendes Werk, ein Kanalwerk, an das periphere Geräte (z.B. Trommel- und Plattenspeicher, Lochkartengeräte, Benutzerkonsolen u.a.) angeschlossen sind, die Arbeit des Rechnerkerns durch ein Signal unterbrechen und dadurch das Organisationsprogramm an die Regie bringen. Da auch Benutzerprogramme und Alarme das Organisationsprogramm aufrufen, kann es immer dann aktiv werden, wenn seine koordinierende Tätigkeit gebraucht wird. Es analysiert dann das aktivierende Signal, beschafft sich eventuell weitere Informationen und ergreift dann die nötigen Maßnahmen. Um über den laufend wechselnden Zustand der Anlage genau informiert zu sein, führt das Organisationsprogramm zahlreiche Listen, die es bei seiner Arbeit stets auf dem neuesten Stand hält.

Vielleicht läßt sich die Wirkungsweise des Organisationsprogramms am besten durch die Darstellung eines konkreten Betriebsablaufs bei einem hypothetischen Teilnehmerrechen-

system veranschaulichen, wobei eine ganze Reihe von Vereinfachungen vorgenommen wird, was für unsere Zwecke aber unwesentlich ist.

Man stelle sich also einen schnellen Rechner vor, der einige hunderttausend Operationen in der Sekunde durchführen kann. Er verfüge über einen Arbeitsspeicher, in dem ein Teil des Organisationsprogramms und einige Benutzerprogramme Platz finden. Über zahlreiche Kanalwerke seien Benutzerkonsolen, Lochkartengeräte, Drucker, Magnetbandeinheiten, Trommel- und Plattenspeicher angeschlossen. Eine Anzahl von Individuen sei berechtigt, über Bedienungskonsolen die Rechenanlage zu benutzen. Ihre Benutzerkennzeichen und Vermerke über ihre besonderen Berechtigungen seien in einer Benutzerliste gespeichert. Programme und Daten einiger dieser Benutzer seien bereits auf dem Plattenspeicher gespeichert. Listen des Organisationsprogramms ermöglichen das Auffinden dieser Informationen. Auf dem Kartenleser liege ein Stapel von nicht sehr dringlichen Programmen, die nebenher ausgeführt werden sollen. In einer weiteren Liste seien Speicherbereiche registriert und dort sei auch vermerkt, von wem sie gerade belegt sind. Auch von den vorhandenen peripheren Geräten ist immer bekannt, wer sie gerade benutzt. Auch die gerade an der Anlage arbeitenden Benutzer sind registriert. Beim Betrieb der Anlage könnte sich nun etwa folgendes abspielen.

An einer bisher freien Konsole hat sich ein Benutzer niedergelassen, einen Dialog eröffnet und seinen Namen geschrie-

ben. Wenn er ein Kommando vervollständigt hat, geht ein Signal an den Rechnerkern, das zu einer Unterbrechung des dort gerade laufenden Programms führt. Das Organisationsprogramm erhält nun die Regie und sein Programmteil, der für die Benutzerverwaltung zuständig ist, prüft nun an Hand seiner Benutzerliste, ob der neue Benutzer zum Kreis der Berechtigten gehört oder nicht. Ist er berechtigt zur Benutzung der Anlage, so registriert das Organisationsprogramm in seiner Geräteliste, daß die entsprechende Konsole von diesem Benutzer belegt ist und beauftragt den Kanal, auf dieser Konsole eine entsprechende Mitteilung auszugeben. Dann wählt es aus der Menge der rechenbereiten Benutzerprogramme eines aus, veranlaßt gegebenenfalls, daß es von einem Hintergrundspeicher in den Arbeitsspeicher geladen wird, teilt ihm ein Rechenzeitquantum zu, nach dessen Ablauf das Programm spätestens wieder unterbrochen wird, indem ein Wecker gestellt wird, und gibt den Rechnerkern für dieses Programm frei. Nach Ablauf der voreingestellten Frist gibt der Wecker ein Signal, das die Zentraleinheit bei der Bearbeitung des Programms unterbricht und wieder das Organisationsprogramm auf den Plan ruft. Dieses stellt fest, daß ein Signal vom Wecker eingetroffen ist und daß der Rechnerkern einem anderen Programm zugewiesen werden muß. Falls notwendig, wird nun der Abtransport des unterbrochenen Programms auf einen Hintergrundspeicher veranlaßt, von dem es zur Fortsetzung dann später wieder geholt werden muß. Aus den anderen rechenbereiten

Aufträgen wählt nun das Organisationsprogramm gemäß der von ihm verfolgten Politik einen zur Bearbeitung aus, läßt ihn notfalls vom Hintergrundspeicher holen, stellt wieder den Wecker und übergibt die Regie an das neue Programm. Dieses Programm benötige jetzt Daten, die schon auf dem Plattenspeicher stehen mögen. Daher ruft es das Organisationsprogramm auf. Dieses erkennt an den vom Programm mitgelieferten Informationen, welcher Bereich vom Plattenspeicher gebraucht wird und gibt dem zuständigen Kanalwerk den Auftrag, diesen Bereich zu übertragen. Da das die Daten anfordernde Programm ohne diese Daten zunächst nicht weiterrechnen kann, wählt das Organisationsprogramm in der oben geschilderten Weise einen neuen Auftrag zur Bearbeitung aus, dem es nun den Rechnerkern übergibt. Dieses möge nun ebenfalls Daten vom Plattenspeicher anfordern. Da der entsprechende Kanal aber noch vom vorhergehenden Auftrag belegt ist, reiht das Organisationsprogramm diese weitere Anforderung in eine Warteschlange ein und wählt erneut einen Auftrag zur Bearbeitung durch den Rechnerkern aus. Hat das Kanalwerk die angeforderten Daten des ersten Auftrages endlich in den Arbeitsspeicher gebracht, so unterbricht es durch ein Signal den Rechnerkern in der Bearbeitung des eben laufenden Programms und bringt dadurch wieder das Organisationsprogramm an die Regie. Dieses prüft, woher das Signal kam und stellt mit Hilfe seiner Listen fest, für welchen Auftrag die Daten bestimmt sind. Dann erhält das Kanalwerk den Transportbefehl für die Daten des zweiten Auftrags. Der Auftrag, der zuerst unterbrochen worden war,

weil er noch auf seine Daten warten mußte, kann jetzt wieder weiterbearbeitet werden.
Das Organisationsprogramm wird außerdem durch Signale aufgerufen, die Alarmmeldungen (z.B. von der Hardware) beinhalten. Derartige Alarme können sofortige Reaktionen erfordern. Beim gleichzeitigen Eintreffen von Unterbrechungssignalen werden solche Signale mit Vorrang behandelt. Sonst aber werden Unterbrechungen, die während des Laufes bestimmter kritischer Teile des Organisationsprogramms eintreffen, verzögert, so daß ein Verlust an Information und eine Verwirrung der Steuerung vermieden wird.
Zusammenfassend kann man sagen:
Das Organisationsprogramm empfängt Signale und trifft in Abhängigkeit von diesen Signalen und dem durch den Inhalt seiner Listen definierten Zustand des Systems Entscheidungen, die im allgemeinen zu einem neuen Zustand des Systems führen.

Es erhebt sich nun die Frage, was man unter einem guten Organisationsverfahren verstehen soll. Eine Bewertung ergibt sich dabei einerseits dadurch, daß jede Verzögerung eines Benutzerprogramms unerwünscht ist und daher in geeigneter Form als Schaden zu betrachten ist. Andererseits ist auch die Nichtausnutzung von zuteilbaren Maschinenkomponenten ein Verlust. Ebenso muß man die Rechenzeit, die nur für die Verwaltungstätigkeit des Organisationsprogramms verbraucht wird, als Schaden ansehen.
Bei der Beurteilung eines Zuteilungsalgorithmus müssen diese Gesichtspunkte in vernünftiger Weise gegeneinander abgewogen werden.

1.5 Zuteilungsalgorithmen einiger realisierter Teilnehmersysteme

Bevor im nächsten Abschnitt einige Bemerkungen zu bisher veröffentlichten theoretischen Arbeiten zur Betriebsmittelverteilung gemacht werden, sollen hier die Zuteilungsalgorithmen einiger realisierter Teilnehmerrechensysteme skizziert werden, um auch praktische Lösungsversuche dieses Problems kurz vorzustellen.

Sehr einfach ist der Zuteilungsalgorithmus in dem bei Bolt, Beranek und Newman entwickelten System für eine PDP-1 (vgl. McCarthy, Boilen, Fredkin u. Licklider 1963). Bei diesem können maximal fünf Benutzer gleichzeitig über Konsolen an der Anlage arbeiten. Als Hintergrundspeicher dient eine Trommel. Ein rechenbereites Benutzerprogramm wird zunächst für maximal 140 Millisekunden bearbeitet. Ist es bis dahin noch rechenbereit, d.h. noch nicht vollständig erledigt, und sind noch weitere rechenbereite Programme vorhanden, so wird in einem bestimmten Zyklus das nächste bereite Programm auf der Trommel ausgewählt, das eben noch bearbeitete Programm hinausgeschrieben und zugleich das neue geladen. Nach 140 msec wiederholt sich dieser Vorgang, so daß alle fünf Benutzer reihum bedient werden. Vor Ablauf des Zeitquantums erfolgt ein Programmaustausch, wenn das Programm in einen nicht rechenbereiten Zustand gerät, z.B. weil es auf die Vollendung eines Ein/Ausgabevorganges warten muß.

Bei größeren Systemen wird auch der Zuteilungsalgorithmus komplizierter. Am Massachusetts Institute of Technology (MIT), wurde ein komplexeres System unter dem Namen CTSS (Compatible Time Sharing System) konstruiert. Der Zuteilungsalgorithmus arbeitete dort wie folgt (Corbatô 1963): Jedem Programm wird bei seiner Ankunft im System eine Priorität zugeteilt, die nach der folgenden Formel berechnet wird:

$$l_0 = [\log_2([w_p/w_q] + 1)],$$

worin w_p die Anzahl der Worte bedeutet, die das Programm im Speicher benötigt, w_q die Anzahl der Worte ist, die während eines Zeitquantums q (ungefähr 200 msec) vom Arbeitsspeicher zum Hintergrundspeicher und zurück transportiert werden können; [a] ist die größte Zahl, die kleiner oder gleich a ist. Die Programme der Priorität 1 bilden jeweils eine eigene Warteschlange. Neu ankommende Programme werden am Ende der entsprechenden Warteschlange eingereiht. Bei der Abarbeitung werden zunächst die Warteschlangen mit der kleinsten Prioritätszahl 1 abgefertigt; und zwar erhält zunächst das erste Programm der Warteschlange 2^1 Zeitquanten. Ist es nach Ablauf dieser Zeit nicht beendet, so wird es am Ende der Warteschlange mit der Priorität 1+1 eingereiht. Ist während der Abarbeitung eines Programms der Priorität 1 ein Programm der Priorität 1' < 1 eingetroffen, so wird der gerade bearbeitete Benutzer an das Ende der 1-ten Warteschlange gesetzt und die Bearbeitung beginnt auf der Stufe 1'.

Die Speicherplatzforderungen eines Programms können sich während seines Laufes natürlich ändern. Dann wird seine Priorität auf folgende Weise korrigiert:
Das Programm habe zunächst w_p Worte belegt und werde in der Priorität 1 bearbeitet, wobei sich seine Speicherplatzanforderung auf w_p'' Worte ändere. Dann wird es am Ende der Warteschlange der Priorität 1" eingereiht, wobei gilt

$$1'' = 1 + [\log_2(w_p''/w_p)].$$

Im Vergleich zu neueren Systemen sind die eben beschriebenen Zuteilungsalgorithmen noch sehr einfach. Als Beispiel für einen neueren Algorithmus sei das von Telefunken für das Teilnehmerrechensystem BS 3 auf dem Rechner TR 440 benutzte Verfahren stark vergröbert skizziert (vgl. TR 440 - Teilnehmerbetriebssystem BS 3 - Einführung, 1970).

An der Vergabe der Betriebsmittel sind hier zwei Instanzen beteiligt, der "Systemkern", der die physikalische Vergabe vornimmt, und die "Kontrollfunktion", die die Betriebsmittelvergabe organisatorisch vorbereitet.

Bei der physikalischen Vergabe des Rechnerkerns sind die Verhältnisse relativ einfach. Dem "Systemkern" als dem Verwalter stehen nicht die Benutzeraufträge selbst gegenüber, sondern eine Reihe von "Akteuren", die zum Teil Benutzeraufträge verwalten können. Diesen Akteuren ist aufgrund ihrer betrieblichen Eigenschaften eine Priorität fest zugeordnet (wobei hier große Zahlen einen hohen Grad der Bevorzugung bedeuten).

Die Benutzeraufträge können in zwei verschiedenen Arten auftreten: als Abschnitte oder als Gespräche. Bei den Abschnitten ist der Benutzer nach der Abgabe des Auftrags an einer weiteren Einflußnahme auf den Gang der Bearbeitung nicht mehr interessiert, während er bei Gesprächen von seinem Bedienungsplatz aus in den Bearbeitungsvorgang eingreifen können möchte. Für die prinzipielle Abwicklung der Benutzeraufträge, sei es in der Form von Gesprächen oder als Abschnitte, sind besondere Akteure, die Gesprächs- und die Abschnittsabwickler zuständig, von denen es zusammen 7 gibt. Die niedrigeren Prioritäten werden dabei von den Abschnittsabwicklern, die höheren von den Gesprächsabwicklern belegt. Die Anzahl der Abwickler des jeweiligen Typs wird von der Rechenzentrumsleitung eingestellt.
Die einzelnen Akteure können sich in verschiedenen Zuständen befinden. Sie können "rechenwillig" sein und sich um die Zuteilung des Rechnerkerns bewerben (die Warteschleife, die die niedrigste Priorität besitzt, ist immer rechenwillig); es können aber auch Akteure "nichtrechenwillig" sein, weil sie z.B. noch auf die Vollendung eines Ein/Ausgabevorgangs warten müssen oder überhaupt kein Auftrag für sie vorliegt. Der Systemkern wird nun durch folgende Ereignisse zu einer Neuvergabe des Rechenwerks veranlaßt: durch Unterbrechungsforderungen, die von peripheren Geräten kommen (Eingriffe), bestimmte Alarme, ein Weckersignal oder die Ausführung gewisser Befehle im gerade laufenden Programm. Die Zuteilungsstrategie des Systemkerns ist nun sehr einfach: er übergibt den Rechnerkern dem rechenwilligen Akteur mit der höchsten

Priorität.

Bei der Vorbereitung der Vergabe der Betriebsmittel durch die Kontrollfunktion werden die für die Auftragsbearbeitung verfügbaren Betriebsmittel mit dem entstehenden Bedarf abgestimmt. Dem Systemkern gegenüber ist die Kontrollfunktion selbst ein Akteur, der durch das Eintreffen des Weckeralarms nach festen Zeitintervallen (etwa 600 msec) den Zustand "rechenwillig" annimmt. Erhält sie den Rechnerkern, so werden Betriebsmittel für Gespräche und Abschnitte auf die folgende Weise verplant. Zunächst werden die Gespräche betrachtet, und zwar zyklisch von der niedrigsten Konsolennummer zur höchsten aufsteigend, sofern noch ein Gesprächsabwickler frei ist. Für das in diesem Zyklus nächste Gespräch wird geprüft, ob seine Anforderungen an Platten-, Trommel- und Kernspeicherplatz erfüllt werden können. Reicht der Platz nicht aus, so wird die Planung für dieses Gespräch abgebrochen. Anderenfalls wird der Speicherplatz abgebucht, das Gespräch wird dem freien Abwickler zugeteilt und dieser "rechenwillig" gesetzt. Ist noch ein Gesprächsabwickler frei und in dem für Gespräche vorgesehenen Kernspeicherbereich noch Platz, so wird versucht, das im Konsolzyklus nächste Gespräch einzuplanen. Nach den Gesprächen werden die Abschnitte eingeplant. Im Gegensatz zu den Gesprächen gibt es wichtigere und weniger wichtige Abschnitte. Jedem Abschnitt ist deshalb eine Gewichtszahl G zugeordnet, wobei G zwischen 1 und 65535 liegt und die kleineren Zahlen größeres Gewicht bedeuten.

Bei Neuankunft eines Abschnitts wird sein Gewicht nach einer von der Rechenzentrumsleitung vorgebbaren Formel aus den Benutzerangaben und -kenndaten berechnet. Im Laufe der Zeit, die ein Abschnitt im System verbringt, wird seine Gewichtszahl kleiner, der Abschnitt also wichtiger.
Jedem der für die Abschnittsbearbeitung zuständigen Akteure (Abschnittsabwickler) wird von der Rechenzentrumsleitung ein Gewichtszahlenintervall zugeordnet. Für alle Abschnitte mit Gewichtszahlen aus diesem Intervall ist dann dieser Abwickler zuständig.
Die Kontrollfunktion sucht nun beim wichtigsten Abschnitt beginnend der Reihe nach die Abschnitte, die auf die Zuteilung eines Abwicklers warten. Gibt es unter den für das Gewicht eines solchen Abschnitts zuständigen Abwicklern einen freien und sind die Speicherplatzforderungen dieses Abschnitts erfüllbar, so wird der Speicherplatz abgebucht, der Abschnitt dem Abwickler zugewiesen und dieser "rechenwillig" gesetzt. Bei der nächsten Rechnerkernvergabe wird der Systemkern die neuen Verhältnisse dann berücksichtigen.

Diese sehr groben Skizzen zeigen, daß die Konstrukteure realisierter Teilnehmerrechensysteme recht verschiedene Verfahren zur Lösung der Zuteilungsprobleme gewählt haben. Auch die neuesten Systeme enthalten noch Möglichkeiten, die Verfahren je nach den Bedürfnissen eines speziellen Rechenzentrums zu verändern. Es liegt nun die Frage nahe, ob ein bestimmter Algorithmus in einer Anlage "gut" ist oder ob man nicht vielleicht "bessere" Verfahren finden

kann. Zur Zeit ist man vermutlich von einer endgültigen Beantwortung dieser Frage schon deshalb noch weit entfernt, weil es kein allgemein akzeptiertes Gütekriterium gibt, das in der Menge der möglichen Zuteilungsalgorithmen eine lineare Ordnung schafft (vgl. z.B. Dorn 1969). Es besteht daher ein starkes Interesse an theoretischen Modellen, die Erkenntnisse zu diesem Problem erhoffen lassen.

1.6 Theoretische Arbeiten über Zuteilungsalgorithmen

Die meisten Autoren, die sich mit analytischen Modellen für Teilnehmerrechensysteme beschäftigen, gehen von der Tatsache aus, daß eine Reihe von Benutzern in zeitlichen Abständen, die dem Zufall unterworfen sind, beim System eintreffen und vom Rechensystem für eine gewisse Zeit, die aus der Sicht des Systems ebenfalls eine Zufallsgröße ist, bedient werden wollen. Da häufig mehrere Benutzer gleichzeitig Dienste wünschen, bildet sich vor dem Rechnerkern eine Warteschlange, die gemäß einer gegebenen Warteschlangendisziplin abzuarbeiten ist. Eine derartige Situation wird auch in der Theorie der Warteschlangen betrachtet, die in den letzten fünzig Jahren ausgehend von der Untersuchung von Telefonzentralen intensiv studiert wurde (vgl. Erlang 1918, Cox u. Smith 1961, Riordan 1962, Takacs 1962, Gnedenko und Kovalenko 1966 u.a.). Neu gegenüber den früher betrachteten Modellen ist aber die Möglichkeit, daß man die Bearbeitung mehrerer Aufgaben in einem "Zeitscheibenverfahren" durchführen kann; dabei wird jede Aufgabe für ein Zeitquantum lang bedient und dann zugunsten einer anderen Aufgabe unterbrochen. Im Unterschied zu den Unterbrechungen bei Prioritätsstrategien erfolgen hier die Unterbrechungen auch, ohne daß eine neue Aufgabe im System eingetroffen ist. Die Warteschlangenmodelle für Teilnehmerrechensysteme müssen nun diese neue Möglichkeit untersuchen. J.M.McKinney gibt 1969 in den Computing Surveys einen Überblick über die wesentlichen bis zu dem Zeitpunkt untersuchten Modelle und die ge-

wonnenen Resultate, die sich unter anderem auf die Erwartungswerte der Verweilzeit eines Auftrags im System (zum Teil in Abhängigkeit von der geforderten Rechenzeit), die Reaktionszeit des Systems und die mittlere Warteschlangenlänge beziehen. Einzelheiten können wir hier übergehen. Im Literaturverzeichnis ist eine ganze Reihe diesbezüglicher Arbeiten aufgeführt.

Der Herausgeber W.S.Dorn bemerkt im gleichen Heft in seinem Kommentar zur Arbeit von Mc Kinney: "After reading this survey, one cannot fail to be struck by the primitiveness of all the models which have been studied to date and the paucity of results of any kind ..... Clearly, much work remains to be done in the modeling of time-sharing systems. The mathematical models described in the paper are already rather complex, but before truly useful results will be forthcoming, the models need to become even more sophisticated and, consequently, more difficult to analyse!"

In der Tat sind alle bekannten Warteschlangenmodelle nicht geeignet, als allgemeine Modelle für Teilnehmerrechensysteme zu dienen. Gemeinsam ist ihnen allen der vorwiegend deskriptive Charakter. Schon zu Beginn der Untersuchung werden die Warteschlangendisziplin und die Bedienungsweise festgelegt. Über das Verhalten des so gewonnenen Modells erhält man dann unter einer Reihe von Voraussetzungen sehr detaillierte Aussagen, die zur Beurteilung des betreffenden Systems von großem Wert sind. Will man jedoch ein Modell mit einer anderen Warteschlangendisziplin zum Vergleich

heranziehen, so muß man in der Regel die Analyse des veränderten Modells ganz neu durchführen, sofern sich die Änderung nicht nur auf einzelne Parameter bezieht, deren Einfluß zu übersehen ist. Muß eine Entscheidung zwischen zwei oder mehr Modellen getroffen werden, so wird diese in der Regel mehr intuitiv gefällt, indem man mit Hilfe der Kenntnisse, die einem zur Verfügung stehen, gefühlsmäßig die verschiedenen, sich oft widersprechenden Planungsziele gegeneinander abwägt. In dieser Situation wäre es nützlich, wenn man alle praktisch sinnvollen Möglichkeiten zum Beispiel für die Festlegung des Zuteilungsalgorithmus betrachten und ein rationales Entscheidungsverfahren aufgrund von Kosten und Nutzen anwenden könnte.

Den gleichen Nachteil, daß sie schon von Anfang an zu stark fixiert sind, haben auch die Modelle in Simulationsstudien (vgl. z.B. Blake u. Gordon 1964, Fine u. McIsaac 1966, Penny 1966, Nielsen 1967, Seaman u. Soucy 1969, Kleinrock 1970, Kümmerle 1970 und andere), die an veränderte Bedingungen noch schwerer anzupassen sind.

Das Problem einer optimalen Zeitzuteilung wird nur von wenigen Autoren betrachtet, auf deren Arbeiten wir an späterer Stelle näher eingehen werden.

2. GRUNDLEGENDE BEGRIFFE AUS DER THEORIE DER STOCHASTISCHEN ENTSCHEIDUNGSMODELLE

2.1 Vorbemerkungen

Die Überlegungen im vorhergehenden Kapitel haben das Unterbrechungskonzept und ein Organisationsprogramm als unentbehrliche Bestandteile eines Teilnehmerrechensystems herausgestellt. Im nächsten Kapitel werden diese Beobachtungen zu einem allgemeinen mathematischen Modell für den Betriebsablauf in Teilnehmerrechensystemen präzisiert, das in enger Beziehung zu den stochastischen Entscheidungsmodellen im Sinne von Hinderer (1970) steht. Zuvor sollen deshalb in diesem Kapitel die wichtigsten Begriffe aus der Theorie dieser Modelle vorgestellt werden.

Die stochastischen Entscheidungsmodelle sind in den vergangenen zehn Jahren intensiv studiert worden, wie die Fülle der zu diesem Thema erschienenen Arbeiten zeigt. Einige Grundideen ihrer Theorie treten wohl zuerst in den Arbeiten A.Wald's zur Sequentialanalyse auf. Sie wurden von R.Bellman weiterentwickelt und unter der Bezeichnung "dynamische Programmierung" auf viele, vorwiegend deterministische Probleme angewandt (vgl.z.B. Bellman 1956, Bellman u. Dreyfus 1962).

Pionierarbeit auf dem Gebiete der stochastischen dynamischen Programmierung, der stochastischen Entscheidungsmodelle, hat vor allem R.A.Howard (1960/1965) mit der Behand-

lung endlicher Markovscher Modelle geleistet.
Besonders wichtig für die Entwicklung der mathematischen Grundlagen der Theorie allgemeinerer Modelle waren die Arbeiten von Blackwell (1965) und Dynkin (1965). Blackwell behandelte stationäre Modelle, bei denen also die Wahrscheinlichkeitsgesetze nicht von der Zeit abhängen, mit allgemeinen Zustands- und Aktionenmengen, Dynkin dagegen betrachtete nichtstationäre Modelle, allerdings nur mit abzählbarem Zustandsraum. Hinderer (1967) hat ein allgemeines nichtstationäres Modell konstruiert, das auch überabzählbare Zustands- und Aktionenmengen enthalten kann. Er hat damit in gewisser Hinsicht eine Synthese der Arbeiten von Dynkin und Blackwell erreicht.

Die stochastischen Entscheidungsmodelle stehen in enger Beziehung zu den 1935 von Onicescu und Mihoc eingeführten "Ketten mit vollständigen Bindungen" (vgl. Onicescu und Mihoc 1935, 1936, Ciucu und Theodorescu 1960). Diese und ihre Verallgemeinerungen haben eine große Bedeutung für die Theorie von Lernmodellen bekommen (vgl. z.B. Karlin 1953, Theodorescu 1960, Iosifescu 1963, Iosifescu und Theodorescu 1969).
Es soll hier nicht der Versuch unternommen werden, einen zusammenfassenden Überblick über die vorhandene Literatur und die bisher erzielten Resultate zu geben, zumal die verschiedenen Arbeiten wegen der Verschiedenartigkeit der untersuchten Probleme und der gemachten Voraussetzungen nur sehr schwer miteinander verglichen werden können. Hierfür

sei auf die Literatur verwiesen. Übersichten enthalten zum Beispiel Sirjaev (1967) und Hinderer (1970).
Die stochastischen Entscheidungsmodelle werden sich bei uns als der allgemeine Rahmen für die Frage nach "guten" Steueralgorithmen für neuere Datenverarbeitungsanlagen ergeben. Wir wollen daher diese Modelle nur unter diesem Gesichtspunkt betrachten. Insbesondere werden keine neuen Sätze hergeleitet.
Die Theorie der stochastischen Entscheidungsmodelle ist recht kompliziert. So sind für gegebene Wahrscheinlichkeitsgesetze, Kosten und Politikenmengen mehr als zwanzig Optimalitätsbegriffe formuliert worden (vgl. Hinderer 1970). Da die Betrachtung der Datenverarbeitungsanlagen mit einiger Konsequenz zu diesen Modellen geführt hat, ist die Vermutung berechtigt, daß die Schwierigkeiten bei einer allgemeinen theoretischen Behandlung der Probleme beim Betrieb solcher Anlagen außer bei den mehr praktischen Schwierigkeiten auch in der Kompliziertheit der zugrundeliegenden mathematischen Struktur liegen und daß auch deshalb noch kein allgemeines Modell bekannt ist.

Wir werden im vierten Kapitel einige dieser Schwierigkeiten und ihre Gründe sehen. Weiter sollen dort einige allgemeine Sätze zitiert werden, die einen für unser Thema relevanten Inhalt haben. Schließlich werden auch Fälle angegeben, in denen eine explizite Konstruktion optimaler Politiken möglich ist.

2.2 Definition stochastischer Entscheidungsmodelle

Im folgenden schließen wir uns eng an die Bezeichnungen von Hinderer (1967) an.

Ein stochastisches Entscheidungsmodell ist ein Quintupel $[\{(Z_t, \mathcal{Z}_t) \mid t=0,1,\ldots\}, \{(A_t, \mathcal{A}_t) \mid t=1,2,\ldots\}, \{q_t \mid t=0,1,\ldots\}, \{g_t \mid t=0,1,\ldots\}, D]$ von Größen, die die folgende Bedeutung haben:

1. $(Z_t, \mathcal{Z}_t)$ ist für jedes t ein Meßraum, der Zustandsraum,
2. $(A_t, \mathcal{A}_t)$ ist für jedes t ein Meßraum, der Aktionenraum.

Von den σ-Algebren fordern wir, daß sie die Einpunktmengen enthalten. Um die übrigen Größen erklären zu können, müssen wir zuvor die Mengen H_t der Vorgeschichten zur Zeit t definieren. Das geschieht durch $H_0 := Z_0$ für t=o und

$$H_t := Z_0 \times A_1 \times Z_1 \times \ldots \times A_t \times Z_t \quad \text{für } t=1,2,\ldots .$$

Über H_t haben wir die σ-Algebren $\mathcal{H}_0 := \mathcal{Z}_0$ für t=o und

$$\mathcal{H}_t := \mathcal{Z}_0 \times \mathcal{A}_1 \times \mathcal{Z}_1 \times \ldots \times \mathcal{A}_t \times \mathcal{Z}_t \quad \text{für } t=1,2,\ldots .$$

Mit H bezeichnen wir die Produktmenge $Z_0 \times \prod_{i=1}^{\infty} (A_i \times Z_i)$.

Die zugehörige Produkt-σ-Algebra werde mit $\mathcal{H}$ bezeichnet.

3. q_t ist für jedes t eine Übergangswahrscheinlichkeit von $H_t \times A_{t+1}$ nach $\mathcal{Z}_{t+1}$. q_t heißt das Bewegungsgesetz des Entscheidungsmodells im Zeitintervall (t,t+1]. Dabei verstehen wir wie üblich unter einer Übergangswahrscheinlichkeit p von dem Meßraum $(A, \mathcal{A})$ in die σ-Algebra $\mathcal{B}$ eine Funktion

$$p : A \times \mathfrak{B} \to [0,1]$$

mit den Eigenschaften

a) für festes $a \in A$ ist $p(a;B)$ eine Wahrscheinlichkeit auf $\mathfrak{B}$;

b) für festes $B \in \mathfrak{B}$ ist $p(a;B)$ eine $\mathfrak{A}$-meßbare Funktion auf A (vgl. Neveu 1969, p.95).

4. g_t ist für jedes t eine auf H_{t+1} definierte, reellwertige $\mathfrak{H}_{t+1}$-meßbare Funktion, die als Gewinnfunktion im Intervall $(t,t+1]$ bezeichnet wird.

5. Eine Politik ist eine Folge $\pi = \{\pi_t \mid t=0,1,\ldots\}$ von Übergangswahrscheinlichkeiten π_t von H_t nach $\mathfrak{A}_{t+1}$. Die Menge D der zulässigen Politiken ist eine nichtleere Teilmenge der Menge S aller Politiken.

Aus der Sicht der Anwendung ist es wichtig zu bemerken, daß die Übergangswahrscheinlichkeiten q_t $(t=0,1,\ldots)$ als fest vorgegeben und bekannt vorausgesetzt werden, während die Politiken π aus D frei gewählt werden können. Für jede Politik π und jede Ausgangsverteilung p_0 auf $\mathfrak{Z}_0$ definiert ein stochastisches Entscheidungsmodell einen stochastischen Prozeß, d.h. ein Wahrscheinlichkeitsmaß $P_{p_0,\pi}$ auf $(H,\mathfrak{H})$. Man interessiert sich für eine optimale Wahl der Politik π; dazu werden wir im übernächsten Abschnitt eine Reihe von Optimalitätsbegriffen einführen.

2.3 Einige bekannte Hilfssätze

Um spätere Zitate zu erleichtern, stellen wir hier einige wichtige Eigenschaften von Wahrscheinlichkeiten zusammen, die mit Hilfe von Übergangswahrscheinlichkeiten konstruiert werden. Dabei verstehen wir unter einer nichtnegativen Zufallsvariablen X über $(E, \mathcal{F})$ eine $\mathcal{F}$- $\bar{\mathcal{B}}_+$-meßbare Abbildung $X : E \to \bar{R}_+$ ($\bar{R}_+$:= Menge der nichtnegativen, durch $+\infty$ kompaktifizierten reellen Zahlen; $\bar{\mathcal{B}}_+$ System der Borelschen Mengen in $\bar{R}_+$).

Satz 1 (Neveu 1969, p.95):

Seien $(E_1, \mathcal{F}_1)$ und $(E_2, \mathcal{F}_2)$ zwei meßbare Räume. P_1 sei eine Wahrscheinlichkeit auf $(E_1, \mathcal{F}_1)$ und P_2^1 eine Übergangswahrscheinlichkeit von $(E_1, \mathcal{F}_1)$ in $\mathcal{F}_2$.

a) Dann existiert eine eindeutig bestimmte Wahrscheinlichkeit P auf $(E_1 \times E_2, \mathcal{F}_1 \times \mathcal{F}_2)$ so, daß

$$P(F_1 \times F_2) = \int_{F_1} P_1(dx_1)\, P_2^1(x_1; F_2) \quad \text{für } F_1 \in \mathcal{F}_1, F_2 \in \mathcal{F}_2.$$

b) Für jede nichtnegative Variable X zum Wahrscheinlichkeitsfeld $(E_1 \times E_2, \mathcal{F}_1 \times \mathcal{F}_2, P)$ ist die Funktion

$$Y(x_1) = \int_{E_2} P_2^1(x_1; dx_2)\; X(x_1, x_2)$$

überall definiert, $\mathcal{F}_1$-meßbar auf E_1 und nichtnegativ. Ferner gilt

$$\int_{E_1 \times E_2} X\, dP = \int_{E_1} P_1\ (dx_1) \int_{E_2} P_2^1(x_1; dx_2)\, X(x_1, x_2).$$

Für abzählbar unendliche Produkträume gilt der auf Ionescu Tulcea 1949 zurückgehende Satz:

Satz 2 (Neveu 1969, p.194):

Sei $\{(E_t, \mathcal{F}_t),\ t=0,1,2,\ldots\}$ eine Folge meßbarer Räume und für jedes t sei $P_{t+1}^{0\ldots t}$ eine Übergangswahrscheinlichkeit von $(\bigtimes_{s=0}^{t} E_s, \bigotimes_{s=0}^{t} \mathcal{F}_s)$ nach $\mathcal{F}_{t+1}$.

a) Dann existiert für jedes $x_0 \in E_0$ genau eine Wahrscheinlichkeit P_{x_0} auf

$$(\Omega, \mathcal{A}) = (\bigtimes_{0}^{\infty} E_t, \bigotimes_{0}^{\infty} \mathcal{F}_t),$$

deren Wert für jeden meßbaren Rechteckzylinder ΠF_t gegeben ist durch

$$P_{x_0}(\Pi F_t) = \chi_{F_0}(x_0) \int_{F_1} P_1^0(x_0;dx_1) \int_{F_2} P_2^{01}(x_0,x_1;dx_2) \ldots .$$

$$\ldots . \int_{F_T} P_T^{0\ldots(T-1)}(x_0,\ldots,x_{T-1};dx_T),$$

wenn nur T so groß gewählt wird, daß $F_t = E_t$ für $t > T$ gilt. (Hierin ist χ_{F_0} die Indikatorfunktion von F_0).

b) Für eine beliebige nichtnegative Variable Y auf $(\Omega, \mathcal{A})$, die nur von den Koordinaten bis T abhängt, gilt:

$$\int_{\Omega} P_{x_0}(d\omega)\, Y(\omega) = \int_{E_1} P_1^0(x_0;dx_1) \int_{E_2} P_2^{01}(x_0,x_1;dx_2) \ldots$$

$$\ldots \int_{E_T} P_T^{0\ldots(T-1)}(x_0,x_1,\ldots,x_{T-1};dx_T)\, Y(x_0,\ldots,x_T).$$

Ferner ist für jede nichtnegative Variable Y auf $(\Omega, \mathcal{A})$ der Ausdruck

$$\int P_{x_0}(d\omega)\, Y(\omega)$$

eine $\mathcal{F}_0$-meßbare Funktion von x_0.

Aus diesen beiden Sätzen erhält man einen Satz, der später die Konsistenz unseres Modells zeigen wird.

Satz 3 (Neveu 1969, p.197):

Es gelten die Voraussetzungen aus Satz 2. P_o sei eine Wahrscheinlichkeit auf $(E_o, \mathcal{F}_o)$.

a) Dann existiert genau eine Wahrscheinlichkeit P auf $(\Omega, \mathcal{A})$, deren Wert auf jedem meßbaren Rechteckzylinder gegeben ist durch

$$P(\prod_t F_t) = \int_{F_o} P_o(dx_o) \int_{F_1} P_1^o(x_o;dx_1) \ldots \int_{F_T} P_T^{o\ldots(T-1)}(x_o,\ldots,x_{T-1};dx_T)$$

sobald T so gewählt wird, daß $F_t = E_t$ für $t > T$.
Diese Wahrscheinlichkeit ist gegeben durch

$$P(A) = \int_{E_o} P_o(dx_o)\, P_{x_o}(A) \quad \text{für } A \in \mathcal{A}.$$

b) Für jede nichtnegative Variable Y auf $(\Omega, \mathcal{A})$ ist die Funktion

$$\int_{\Omega} P_{x_o}(d\omega)\, Y(\omega) \quad \text{von } x_o,$$

welche überall definiert ist, eine Version des bedingten Erwartungswertes $E^{\mathcal{F}_o}(Y)$, der auf $(E_o, \mathcal{F}_o)$ definiert ist.

2.4 Optimalitätsbegriffe

In 2.2 ist ein allgemeines stochastisches Entscheidungsmodell definiert worden. Die "Güte" einer Politik soll über den Erwartungswert des Gesamtgewinns

$$(1) \qquad g = \sum_{t=o}^{\infty} g_t$$

definiert werden. Um die Existenz dieses Erwartungswertes sicherzustellen, sind noch Voraussetzungen an die Funktionen g_t nötig, die wir uns auf ganz H definiert, aber nur von den Koordinaten aus H_{t+1} abhängig denken. Aus der Literatur sind verschiedene solcher Voraussetzungen bekannt (vgl.z.B. Hinderer 1970). Die wichtigsten sind die folgenden, wobei für eine Funktion $f : H \to R$ $\|f\| := \sup_{h \in H} |f(h)|$ sei.

Man sagt, für ein Entscheidungsmodell liegt der im wesentlichen negative Fall (EN) vor, wenn gilt

$$(2) \qquad \sum_t \|g_t^+\| < \infty .$$

Man spricht von im wesentlichen positiven Fall (EP), wenn gilt

$$(3) \qquad \sum_t \|g_t^-\| < \infty .$$

Liegen gleichzeitig der Fall (EP) und der Fall (EN) vor, so spricht man vom konvergenten Fall (C), der also genau dann vorliegt, wenn gilt

$$(4) \qquad \sum_t \|g_t\| < \infty .$$

Der Fall (EN) liegt zum Beispiel vor, wenn alle g_t negativ sind, wenn also nur Schäden auftreten. Sind alle g_t nichtnegativ, so liegt der Fall (EP) vor. Der Fall (C) liegt zum Beispiel dann vor, wenn es Konstante $c \geq 0$ und $\beta \in [0,1)$ gibt, so daß für alle t gilt: $|g_t| \leq c \cdot \beta^t$.

Aus jeder der angegebenen Voraussetzungen folgt die Existenz von g als quasiintegrable Zufallsgröße mit Werten aus $[-\infty,+\infty]$ auf $(H, \ )$ mit einer beliebigen Wahrscheinlichkeit.

Der Erwartungswert von g bezüglich des für das Entscheidungsmodell durch die Anfangsverteilung p_o und die Politik π definierten Wahrscheinlichkeitsmaßes P_π auf $\mathcal{H}$ sei G_π. Es gilt:

$$(5) \qquad G_\pi := E_\pi g = \int g \, dP_\pi ,$$

G_π heißt der <u>erwartete Gesamtgewinn</u>.

Außerdem wird der erwartete Gewinn in den Schritten (n,∞) benötigt, wenn bis zum Zeitpunkt n die Vorgeschichte $h \in H_n$ durchlaufen wurde. Zu einer exakten Definition setzen wir $(E_o, \mathcal{E}_o) = (H_n, \mathcal{H}_n)$, $(E_{2\nu+1}, \mathcal{E}_{2\nu+1}) = (A_{n+\nu+1}, \mathcal{A}_{n+\nu+1})$, $(E_{2\nu+2}, \mathcal{E}_{2\nu+2}) = (Z_{n+\nu+1}, \mathcal{Z}_{n+\nu+1})$, $P^{o\ldots 2\nu}_{2\nu+1} = \pi_{n+\nu}$ und $P^{o\ldots 2\nu+1}_{2\nu+2} = q_{n+\nu}$ für $\nu=0,1,2,\ldots$

Nach (2.3) Satz 1 konstruieren wir dann zu $h \in H_n$ die Wahrscheinlichkeit $P_{h,\pi}$ und definieren

$$(6) \qquad G_{n,\pi}(h) := \int \sum_{t=n}^{\infty} g_t \, dP_{h,\pi} \quad (n=0,1,2,\ldots).$$

$G_{n,\pi}$ ist dann $\mathcal{H}_n$-meßbar und nach (2.3) Satz 3 eine Version des bedingten Erwartungswertes $E_{P_\pi}(\sum_{t=n}^{\infty} g_t \mid \mathcal{H}_n)$.

Der optimale Gewinn im Zeitraum (n,∞) bei Vorliegen der Vorgeschichte $h \in H_n$ ist

$$(7) \qquad G_n(h) := \sup_{\pi \in D} G_{n,\pi}(h).$$

Ein Beispiel von Strauch (1966) zeigt, daß G_n im allgemeinen nicht $\mathcal{B}_n$-meßbar ist.

(8) Weiter sei $\qquad G := \sup_{\pi \in D} G_\pi$.

Man sieht nun die folgenden Beziehungen sofort ein:

$$(9) \qquad G_\pi = \int G_{0,\pi}\, dP_0 \qquad \text{und}$$

$$(10) \qquad G_{n,\pi}(h_n) = \int \pi_n(h_n;da_{n+1}) \int q_n(h_n,a_{n+1};dz_{n+1}) \left[g_n(h_n,a_{n+1},z_{n+1}) + G_{n+1,\pi}(h_n,a_{n+1},z_{n+1})\right]$$

für $n=0,1,2,\ldots$, jede Politik π und $h_n \in H_n$.

Ist M_n die Menge aller meßbaren Funktionen $v:H_n \to \bar{R}$, die im Falle (EP) nach unten, im Falle (EN) nach oben beschränkt sind, so erhält man eine einfachere Schreibweise durch Einführung der Operatoren $L_{n,\pi}:M_{n+1} \to M_n$, die wie folgt definiert seien: Der Wert der Bildfunktion $L_{n,\pi}u$ für $u \in M_{n+1}$ bei $h_n \in H_n$ ist

$$(11) \qquad (L_{n,\pi}u)(h_n) := \int \pi_n(h_n;da_{n+1}) \int q_n(h_n,a_{n+1};dz_{n+1}) \left[g_n(h_n,a_{n+1},z_{n+1}) + u(h_n,a_{n+1},z_{n+1})\right].$$

Die Gleichung (10) nimmt dann die einfache Gestalt an:

$$(12) \qquad G_{n,\pi}(h_n) = (L_{n,\pi}\, G_{n+1,\pi})(h_n) \quad \text{für } n=0,1,2,\ldots$$

(Falls die Wahrscheinlichkeit $\pi_n(h_n;.)$ in einem Punkt a konzentriert ist, schreiben wir statt $L_{n,\pi}$ auch $L_{n,a}$).

Mit Hilfe von $G_{n,\pi}$ wird eine Reihe von Optimalitätsbegriffen eingeführt:

Eine Politik π heißt stark optimal, falls

$$(13) \qquad G_{n,\pi}(h) = \sup_{\delta \in D} G_{n,\delta}(h)$$

für alle n und $h \in H_n$.

Die Tatsache, daß optimale Politiken in diesem starken Sinne häufig nicht existieren, zwingt zur Einführung schwächerer Optimalitätsbegriffe.

Schwächer ist die Forderung, daß eine optimale Politik die optimalen Gewinne nur bis auf eine vorgebbare Schranke erreichen muß.

Sei $\varepsilon = (\varepsilon_n)$ eine Folge von nichtnegativen, reellen Zahlen. Eine Politik π heißt dann stark ε-optimal bezüglich dieser Folge, wenn gilt

$$(14) \qquad G_{n,\pi}(h) \geq \sup_{\delta \in D} G_{n,\delta}(h) - \varepsilon_n$$

für jedes n und $h \in H_n$.

Eine stark optimale Politik ist also für jede Folge ε stark ε-optimal, insbesondere für die aus lauter Nullen bestehende Folge.

Auch stark ε-optimale Politiken existieren häufig nicht. Man schwächt daher die Forderungen weiter ab, indem man (14) nicht mehr für alle h verlangt, sondern nur noch für alle h aus einer Menge mit vorgebbarer Wahrscheinlichkeit, die im allgemeinen auch kleiner als 1 gewählt werden kann.

Sei also $(p_n) =: p$ eine Folge von Wahrscheinlichkeitsmaßen auf $\mathfrak{H}_n$, $\varepsilon = (\varepsilon_n)$ eine Folge nichtnegativer, reeller Zahlen. Eine Politik π heißt dann stark (p,ε)-optimal, falls gilt

(15) $p_n(G_{n,\pi} \geq G_{n,\delta} - \varepsilon_n) = 1$

für alle n und $\delta \in D$.

Eine Abschwächung der starken Optimalität in einer anderen Richtung erhält man, wenn man die Ungleichungen in den angegebenen Definitionen nicht für alle n fordert, sondern nur für n = 0.

Eine Politik π heißt optimal, falls gilt

(16) $G_{o,\pi}(z_o) \geq G_{o,\delta}(z_o)$ für alle $z_o \in Z_o$ und alle $\delta \in D$.

Auch diese Forderung wird abgeschwächt:

Eine Politik π heißt ε-optimal für ein $\varepsilon \geq 0$, wenn gilt

(17 $G_{o,\pi}(z_o) \geq G_{o,\delta}(z_o) - \varepsilon$

für jedes $z_o \in Z_o$ und jede Politik δ aus D.

Eine optimale Politik ist also ε-optimal für jedes ε .

Mit einem Wahrscheinlichkeitsmaß p auf $\mathcal{Z}_o$, also über der Menge der Anfangszustände, lassen sich weitere Optimalitätsbegriffe formulieren.

Eine Politik π heißt im Mittel p-optimal (M-p-optimal, $\bar{p}$-optimal), wenn gilt

(18) $\int G_{o,\pi} dp = \sup_{\delta \in D} \int G_{o,\delta}\, dp.$

Dieser Begriff wird in Hinderer (1970) ausführlich behandelt.

Eine Politik π heißt p-optimal, bezüglich des gegebenen Wahrscheinlichkeitsmaßes p, falls gilt

(19) $p(G_{o,\pi} \geq G_{o,\delta}) = 1$ für $\delta \in D$.

Eine optimale Politik ist p-optimal für jedes Wahrscheinlichkeitsmaß p auf $\mathcal{Z}_o$.

Eine Politik π heißt (p,ε)-optimal für $\varepsilon \geq 0$ und das gegebene Wahrscheinlichkeitsmaß p auf $\mathfrak{Z}_0$, falls gilt

(20) $\quad p(G_{o,\pi} \geq G_{o,\delta} - \varepsilon) = 1$ für $\delta \in D$.

Eine p-optimale Politik ist (p,ε)-optimal für jedes $\varepsilon \geq 0$. Eine ε-optimale Politik ist (p,ε)-optimal für jede Verteilung p auf $\mathfrak{Z}_0$.

Zwei weitere Optimalitätsbegriffe erhält man, wenn die Maximalitätsbedingungen in den beiden vorhergehenden Definitionen nicht mit Wahrscheinlichkeit 1 gelten müssen, sondern eine Ausnahmemenge der Wahrscheinlichkeit $\gamma \geq 0$ zugelassen ist.

Eine Politik π heißt dann (γ,p)-optimal, falls gilt

(21) $\quad p(G_{o,\pi} \geq G_{o,\delta}) \geq 1 - \gamma$ für jedes $\delta \in D$.

Eine Politik heißt (γ,p,ε)-optimal, falls gilt

(22) $\quad p(G_{o,\pi} \geq G_{o,\delta} - \varepsilon) \geq 1 - \gamma$ für $\delta \in D$.

(γ,p,ε)-Optimalität ist offensichtlich die schwächste Art der Optimalität.

Die Begriffe einer (γ,p,ε)-optimalen und der im Mittel p-optimalen Politik gehen wohl auf Hinderer (1967) zurück, die übrigen Begriffe stehen im stationären Fall bei Blackwell (1965).

Andere Optimalitätsbegriffe stützen sich auf die mittleren Gewinne pro Schritt (vgl. Howard (1960, 1965), Derman (1962)). Gegen Ende des 4. Kapitels werden wir für einen Spezialfall auf den mittleren Gewinn eingehen.

3. EIN ALLGEMEINES MATHEMATISCHES MODELL

3.1 Heuristik und mathematische Formulierung des Modells

Das zu konstruierende mathematische Modell soll die Arbeitsweise des Organisationsprogramms einer neueren Datenverarbeitungsanlage beschreiben und die Möglichkeit geben, die Frage nach einem "guten" Organisationsprogramm zu formulieren und unter geeigneten zusätzlichen Voraussetzungen auch zu beantworten. Die Aufgaben und die Funktionsweise des Organisationsprogramms wurden unter technischen Gesichtspunkten bereits oben beschrieben.

Danach benötigt das Organisationsprogramm eine Reihe von Informationen, die den Zustand des Systems definieren, soweit dieser für die Organisation von Bedeutung ist. Diese Informationen sind in gewissen Listen enthalten und beziehen sich einerseits auf die im System gerade aktiven Benutzer, andererseits auf die Ausstattung und Belegung der Anlage. Für jeden Benutzer könnte man etwa seine verbrauchte Rechenzeit, seine Wartezeit, Rechenzeit- und Speicherplatzschranken, seine Dringlichkeitsstufe und anderes berücksichtigen. Von der Anlage wird man wissen müssen, welche Komponenten überhaupt betriebsbereit sind, welche mit Aufgaben beschäftigt und welche noch zu vergeben sind. Bei Time-Sharing-Systemen wird man auch noch die Größe der Zeitvorgaben beachten müssen.

Das System soll zu gewissen diskreten, nicht notwendig äqui-

distanten Zeitpunkten t=0,1,... betrachtet werden. Der Zustand, in dem sich das System zur Zeit t befindet, werde mit s_t bezeichnet. Die Menge der denkbaren Zustände zur Zeit t sei S_t.

Die Abhängigkeit der Mengen S_t von der Zeit t ließe sich leicht eliminieren, indem man zur Vereinigungsmenge aller S_t übergeht; da sie hier aber nicht stört, verzichten wir auf diesen Schritt. Die Mächtigkeit von S_t hängt von den Einschränkungen ab, denen man die zu berücksichtigenden Parameter unterwirft. In der Praxis wird man wohl mit endlichen Mengen auskommen, doch sind bereits für leicht idealisierte Modelle unendliche Mengen erforderlich (siehe z.B. 3.3 und 5.1). Über S_t sei eine σ-Algebra $\mathcal{T}_t$ gegeben, die die einpunktigen Mengen enthalte. Ist S_t höchstens abzählbar, so ist $\mathcal{T}_t$ die Menge $\mathcal{P}(S_t)$ aller Teilmengen von S_t.

Das Organisationsprogramm wird durch das Eintreffen bestimmter Signale aufgerufen, die ihm Informationen über die beim Betriebsablauf eingetretenen Ereignisse vermitteln. Die Signale können etwa Meldungen folgender Art beinhalten: ein Benutzer vom Typ "sehr dringlich" meldet sich über eine Konsole an und teilt gleichzeitig mit, daß er maximal eine reine Rechenzeit der Länge T benötigen wird, daß er eine bestimmte Anzahl von Speicherplätzen wünscht und daß er diese und jene externen Geräte benutzen möchte. Weiter könnte gemeldet werden, daß ein Programm, das eben noch ein Rechenwerk belegt hatte, seinen Lauf mit einem Fehler beendet hat. Oder der Wecker signalisiert, daß ein vorher eingestell-

tes Zeitintervall abgelaufen ist; in einem Time-Sharing-System könnte das bedeuten, daß ein Zeitquantum, das einem Benutzer zugeteilt worden war, zu Ende ist und das Programm daher zugunsten eines anderen Benutzers unterbrochen werden muß. Diese kurze Aufzählung zeigt, an welche Art von Ereignissen gedacht ist.

Die Zeitpunkte t=0,1,2,... in unserem Modell werden durch das Eintreffen der Signale definiert. t=0 gibt den Startzeitpunkt an, t=1 den Zeitpunkt des Eintreffens des ersten Signals, t=2 den des zweiten Signals usw. Dabei wird vorausgesetzt, daß auch bei gleichzeitig eintreffenden Signalen definiert ist, welches als erstes, welches als zweites usw. zu betrachten ist. In Datenverarbeitungsanlagen wird das durch gewisse Vorrangregeln erreicht. Die Zeitdauer τ_t, die zwischen dem (t-1)-ten Zeitpunkt und dem t-ten verstreicht, also die Zeit zwischen den Ankünften des (t-1)-ten Signals und des t-ten Signals, fassen wir mit dem Signal σ_t zusammen zum Ereignis e_t. Die Menge aller denkbaren Ereignisse im t-ten Schritt sei E_t. Über E_t sei eine σ-Algebra $\mathcal{E}_t$ gegeben, die die Einpunktmengen enthalte.

Das Eintreten eines Ereignisses bewirkt, daß das Organisationsprogramm tätig wird. Beim t-ten Ereignis wird dieses eine Aktion o_t aus einer Menge O_t von denkbaren Aktionen wählen. Auf O_t sei wieder eine σ-Algebra $\mathcal{O}_t$ gegeben, die ebenfalls die Einpunktmengen enthalte.

Eine Aktion könnte etwa darin bestehen, daß das Organisationsprogramm einen der anstehenden Benutzeraufträge zur

Bearbeitung auswählt und diesem für eine gewisse Maximalzeit den Rechnerkern übergibt. Andere Aktionen sind die Unterbrechung eines Programms, Zuweisung von Speicherbereichen und Kanalwerken zu peripheren Geräten, Eintragungen, Änderungen und Löschungen in den dem Organisationsprogramm zugänglichen Listen und anderes mehr. Welche Aktionen sinnvoll sind, hängt offenbar vom Zustand des Systems und von den eingetroffenen Meldungen ab.

Ein Wort soll noch zu der vielleicht etwas irreführenden Ausdrucksweise "das Organisationsprogramm wählt eine Aktion aus" gesagt werden. Wenn das Organisationsprogramm installiert ist, hat es keine echten Wahlmöglichkeiten mehr, da der Programmierer dieses Programms die zu ergreifende Aktion in Abhängigkeit von den verfügbaren Informationen bereits festgelegt hat. Die Wahlmöglichkeit hat also eigentlich nur der Programmierer. Wenn hier davon die Rede ist, welche Aktion das Organisationsprogramm ergreifen soll, so ist natürlich gemeint, daß man davon spricht, welche Aktionen der Programmierer vorschreiben soll. Das bedeutet insbesondere:

Ein Optimierungsverfahren, das diese Wahl bezüglich eines Kriteriums optimiert, läuft nicht, wie man vielleicht glauben könnte, während des Betriebes der Rechenanlage ab, sondern vorher beim Entwurf des Organisationsprogramms. Das fertige Organisationsprogramm hat dann nur noch die Aufgabe, den Zustand des Systems und die Art des eingetroffenen Signals festzustellen und die zugehörige optimale Aktion einzuleiten. Mathematisch wird das wie folgt beschrieben:

Aus den Zuständen, die das System bis zu einem bestimmten Zeitpunkt t durchlaufen hat, den bis dahin eingetretenen Ereignissen und den vom Organisationsprogramm veranlaßten Aktionen bestehen die Vorgeschichten zum Zeitpunkt t, die wie folgt definiert seien:

$v_o^\varepsilon := s_o \in S_o$ ist eine Vorgeschichte des ersten Ereignisses, S_o ist die Menge aller Vorgeschichten des ersten Ereignisses mit der σ-Algebra $\mathcal{T}_o$.

Unter einer Vorgeschichte des (t+1)-ten Ereignisses verstehen wir

$$v_t^\varepsilon := (s_o, e_1, o_1, s_1, e_2, \ldots, e_t, o_t, s_t) \quad \text{für } t=1,2,\ldots ,$$

dabei sind $s_i \in S_i$, $e_i \in E_i$ und $o_i \in O_i$ für i=1,2,.. und $s_o \in S_o$.

Die Menge dieser Vorgeschichten ist dann

$$V_t^\varepsilon = S_o \times E_1 \times O_1 \times S_1 \times E_2 \ \ldots \ E_t \times O_t \times S_t \text{ für } t=0,1,2,\ldots .$$

Als σ-Algebren über den Mengen von Vorgeschichten nimmt man die zugehörigen Produkt-σ-Algebren

$$\mathcal{W}_t^\varepsilon = \mathcal{T}_o \times \mathcal{E}_1 \times \mathcal{O}_1 \times \mathcal{T}_1 \times \mathcal{E}_2 \times \ldots \times \mathcal{E}_t \times \mathcal{O}_t \times \mathcal{T}_t \quad \text{für } t=0,1,2,\ldots .$$

Eine Vorgeschichte der t-ten Aktion ist

$$v_t^\omega := (s_o, e_1, o_1, s_1, e_2, \ldots, o_{t-1}, s_{t-1}, e_t) \text{ aus der Menge}$$

$$V_t^\omega := S_o \times E_1 \times O_1 \times S_1 \times E_2 \times \ldots \times O_{t-1} \times S_{t-1} \times E_t$$

mit der Produkt-σ-Algebra

$$\mathcal{W}_t^\omega := \mathcal{T}_o \times \mathcal{E}_1 \times \mathcal{O}_1 \times \mathcal{T}_1 \times \mathcal{E}_2 \times \ldots \times \mathcal{O}_{t-1} \times \mathcal{T}_{t-1} \times \mathcal{E}_t$$

für t=1,2,... .

Eine Vorgeschichte des t-ten Zustandes ist

$v_t^\sigma := (s_o, e_1, o_1, s_1, e_2, \ldots, o_{t-1}, s_{t-1}, e_t, o_t)$ aus der Menge

$V_t^\sigma := S_o \times E_1 \times O_1 \times S_1 \times E_2 \times \ldots \times O_{t-1} \times S_{t-1} \times E_t \times O_t$

mit der Produkt-σ-Algebra

$$\mathfrak{V}_t^\sigma := \mathcal{S}_o \times \mathcal{E}_1 \times \mathcal{O}_1 \times \mathcal{S}_1 \times \mathcal{E}_2 \times \ldots \times \mathcal{O}_{t-1} \times \mathcal{S}_{t-1} \times \mathcal{E}_t \times \mathcal{O}_t$$

für t=1,2,... .

Da die ursprünglichen σ-Algebren die einpunktigen Mengen enthielten, haben auch die angegebenen Produkt-σ-Algebren diese Eigenschaft.

Jetzt müssen noch das stochastische Verhalten des Systems und die Wahl der Aktionen durch das Organisationsprogramm genauer beschrieben werden.

Dazu nehmen wir an, daß die Wahrscheinlichkeiten für das Eintreten der Ereignisse aus $\mathcal{E}_t$ bekannt sind, falls das System die Vorgeschichte v_t^ε durchlaufen hat (t=0,1,...). Diese Verteilung sei durch eine Übergangswahrscheinlichkeit q_t' vom Meßraum $(V_t^\varepsilon, \mathfrak{V}_t^\varepsilon)$ nach $\mathcal{E}_{t+1}$ beschrieben.

Die Tätigkeit des Organisationsprogramms zum Zeitpunkt t besteht darin, eine Aktion o_t aus O_t zu wählen. In der Praxis wird die Wahl von o_t nur von einem kleinen Teil der Vorgeschichte, im Extremfall nur von s_{t-1} und e_t abhängen und deterministisch erfolgen. Trotzdem ist aber die Frage interessant (vgl.z.B. das "working set"-Modell von Denning 1968), ob eine Berücksichtigung eines größeren Teils der Vorgeschichte das Betriebsverhalten des Systems verbessern kann.

Ebenso kann es sinnvoll sein, randomisierte Entscheidungen in Betracht zu ziehen (zum Beispiel dann, wenn die Benutzer aus der Kenntnis eines deterministischen Entscheidungsverfahrens zum Schaden des Gesamtbetriebs für sich selbst Nutzen ziehen können). Wir verstehen daher unter einer Strategie zum Zeitpunkt t eine Übergangswahrscheinlichkeit π'_t von $(V^{\omega}_{t+1}, \mathcal{W}^{\omega}_{t+1})$ nach $\mathcal{O}_{t+1}$ (t=0,1,2,...). Zu jedem Zeitpunkt gibt es eine von der Vorgeschichte abhängige Menge von zulässigen Strategien, aus denen bei der Festlegung des Organisationsprogramms eine gewählt werden kann, z.B. können aus logischen oder technischen Gründen gewisse Aktionen verboten sein. Eine Betriebspolitik $\pi = (\pi'_0, \pi'_1, \ldots)$, und damit das Organisationsprogramm, liegt fest, wenn für jedes t eine zulässige Strategie festgelegt wurde. Die Gesamtheit der zulässigen Betriebspolitiken sei Π.

Wenn das Organisationsprogramm die zum Zeitpunkt t gewählte Aktion ausgeführt hat, geht das System in den neuen Zustand s_t über. Dieser Übergang werde auch durch eine Übergangswahrscheinlichkeit p'_t von $(V^{\sigma}_t, \mathcal{W}^{\sigma}_t)$ nach γ_t bestimmt. Der Zustand s_o beim Start des Systems werde durch die Wahrscheinlichkeit p'_o auf γ_o festgelegt. Der neue Zustand ist im allgemeinen nicht durch die Aktion des Organisationsprogramms determiniert, denn die Aktion kann z.B. darin bestehen, Informationen von außerhalb des Systems zur Festlegung des Zustands hereinzuholen. Die von außen eintreffende Information muß dann aus der Sicht des Systems als zufällig betrachtet werden.

Die Ausgangswahrscheinlichkeit p'_o, die Übergangswahrscheinlichkeiten $p'_1, p_2, \ldots$ und $q'_o, q'_1, \ldots$ legen zusammen mit einer Politik π' das stochastische Verhalten des Systems völlig fest. Die Wahl dieser Bestimmungsstücke in der beschriebenen Weise führt auch nicht zu Widersprüchen. Um das zu beweisen, wenden wir Satz 3 aus (2.3) mit den folgenden Identifizierungen an, wobei für den Augenblick der Meßraum der Betriebsereignisse mit $(E'_t, \mathfrak{E}'_t)$ bezeichnet werde:

$$(E_{3i}, \mathfrak{F}_{3i}) := (S_i, \mathfrak{S}_i),$$
$$(E_{3i+1}, \mathfrak{F}_{3i+1}) := (E'_{i+1}, \mathfrak{E}'_{i+1}) \text{ und}$$
$$(E_{3i+2}, \mathfrak{F}_{3i+2}) := (O_{i+1}, \mathfrak{O}_{i+1}) \text{ für } i=0,1,2,\ldots$$

Die Übergangswahrscheinlichkeiten bei einer festen Politik $\pi' = (\pi'_o, \pi'_1, \pi'_2, \ldots)$ sind:

$$p_1^o := q'_o,$$
$$p_2^{o,1} := \pi'_o,$$
$$p_3^{o,1,2} := p'_1, \text{ und für } i=1,2,\ldots$$
$$p_{3i+1}^{o\ldots(3i)} := q'_i,$$
$$p_{3i+2}^{o\ldots(3i+1)} := \pi'_i \text{ und}$$
$$p_{3(i+1)}^{o\ldots(3i+2)} := p'_{i+1}.$$

Der Satz besagt dann, daß jede Politik π' genau eine Wahrscheinlichkeit $P_{p'_o,\pi'}$ auf $(\bigtimes_{s=o}^{\infty} E_s, \bigtimes_{s=o}^{\infty} \mathfrak{F}_s)$ definiert, die für die Rechteckszylinder die oben beschriebenen Werte annimmt, falls eine Ausgangswahrscheinlichkeit p'_o gegeben ist. Das heuristisch aufgestellte Modell für den Betriebsablauf in einer Datenverarbeitungsanlage ist also widerspruchsfrei.

Mit dem Maß $P_{p_o',\pi'}$ auf dem angegebenen Raum ist ein stochastischer Prozeß definiert. Ein Pfad dieses Prozesses entspricht einem speziellen Betriebsablauf in unserem Modell. Zur Veranschaulichung möge das Entstehen eines solchen Pfades $(s_o,e_1,o_1,s_1,....)$ skizziert werden, wie man es in einer Simulation nachbilden kann: Zunächst wird der Ausgangszustand s_o gemäß der Wahrscheinlichkeitsverteilung p_o' aus S_o durch einen Zufallsmechanismus erzeugt. $q_o'(s_o;B_1)$ ist als Funktion von $B_1 \in \mathcal{E}_1$ eine von s_o abhängige Wahrscheinlichkeit auf $(E_1, \mathcal{E}_1)$, nach der nun ein Betriebsereignis e_1 aus E_1 erzeugt wird. Die Betriebspolitik π' legt durch π_o' die von v_1^{ω} abhängige Wahrscheinlichkeit auf $(O_1, \mathcal{O}_1)$ fest, nach der die Aktion o_1 erzeugt wird. Der neue Zustand s_1 wird nun mit der von v_1^{σ} abhängigen Wahrscheinlichkeit p_1' erzeugt usw.

In einer Simulationsstudie eines solchen Modells könnte man diesen Erzeugungsvorgang für die Pfade in einem Programm nachbilden, indem man zur Realisierung der Zustände, Betriebsereignisse und Aktionen Zufallszahlen- oder Pseudozufallszahlengeneratoren zu den angegebenen Wahrscheinlichkeitsverteilungen heranzieht (vgl. z.B. Lehmann 1967). Genau dieses geschieht in den oben erwähnten Simulationsstudien von Time-Sharing-Modellen aus der Literatur für spezielle Modelle (z.B. Fine und McIsaac 1966, Kleinrock 1970, Kümmerle 1970, Nielsen 1967 u.a.).

3.2 Bewertung von Politiken

Das in 3.1 konstruierte Modell ist zwar geeignet, in sehr allgemeiner Form den Betriebsablauf zu beschreiben, aber ein Ziel dieser Arbeit besteht auch darin, Aussagen zum Problem einer "guten" Festlegung der Betriebspolitik zu machen. Zu diesem Zweck benötigen wir noch eine Bewertung der Betriebspolitiken gemäß folgenden Überlegungen.

Der Anstoß für die Entwicklung von Teilnehmerrechensystemen ging von dem Wunsch aus, einer bestimmten Gruppe von Benutzern gleichzeitig eine möglichst kurze Antwortzeit des Systems zu verschaffen. Man wird also die Verzögerung eines Programms eines solchen Benutzers als einen Schaden betrachten, den wir uns durch eine reelle Zahl ausgedrückt denken. Diese Schäden können für verschiedene Benutzer auch verschieden groß sein. Weiter wird man den Schaden, der durch die Verzögerung eines Programms entsteht, nicht unabhängig von der Vorgeschichte annehmen können. So wird zum Beispiel die erste halbe Sekunde, die ein Benutzer bei einem Gespräch auf eine Antwort warten muß, für ihn nicht besonders schlimm erscheinen. Mit wachsender Wartezeit wird der Benutzer dann immer unwilliger, bis ein Punkt erreicht wird, bei dem er nicht mehr bereit ist, länger zu warten. Im allgemeinen kann man annehmen, daß der Schaden, mit dem wir diesen "Benutzergroll" messen wollen, nicht linear mit der Wartezeit zunimmt, so daß für eine Aussage darüber, welchen Schaden die Verzögerung eines Programms um eine

Zeiteinheit anrichtet, die Kenntnis der bisher schon aufgetretenen Verzögerungen dieses Programms nötig ist. Dieser Schaden hängt also im allgemeinen von der Vorgeschichte ab.

Nicht nur die Wartezeiten der Benutzer sind unerwünscht. Es soll ja auch eine gute Auslastung der Anlage und ihrer Komponenten erzielt werden. Wir haben also weitere Schäden zu verzeichnen, wenn Anlagenteile untätig sind, obwohl Programme vorhanden sind, die sie benutzen könnten.

In Kapitel 1 wurde dargelegt, daß für die Steuerung des Betriebsablaufs in neueren Datenverarbeitungsanlagen ein Organisationsprogramm unentbehrlich ist. Auf der anderen Seite jedoch verbraucht dieses Programm jedesmal, wenn es aufgerufen wird, Rechenzeit, die den eigentlichen Anwendungsprogrammen verloren geht und bei der Bewertung eines Systems als Schaden zu betrachten ist: Eine Rechenanlage, deren einzige Tätigkeit darin besteht, sich selbst zu verwalten, ist völlig nutzlos. Wieviel Rechenzeit dieses Organisationsprogramm in einem realisierten Rechensystem verbraucht, hängt von den Aktionen ab, die es bei seinem Aufruf auszuführen hat. Man wird also von einem "guten" Organisationsprogramm erwarten, daß aufwendige Aktionen möglichst selten durchzuführen sind. Es ist oben schon bemerkt worden, daß häufige Unterbrechungen von Benutzerprogrammen zur Erzielung kurzer Reaktionszeiten mit einem erhöhten Organisationsaufwand verbunden sind und daher möglichst größere Zeitquanten anzustreben sind. Der Schaden hängt also im allgemeinen auch von den Aktionen und den Ereignissen ab.

Aufgrund dieser Überlegungen definieren wir als Schadensfunktionen eine Folge $\{k_t(s_o,e_1,o_1,s_1,\ldots,s_{t+1},e_{t+2})$ für $t=0,1,2,\ldots\}$ von reellwertigen, $\mathfrak{W}^{\omega}_{t+2}$-meßbaren Funktionen. Diese liefern für den Betriebsablauf $(s_o,e_1,o_1,s_1,e_2,\ldots)$ den gesamten Schaden bis zum (N+2)-ten Signal

$$S_N = \sum_{t=o}^{N} k_t(s_o,e_1,o_1,\ldots,s_{t+1},e_{t+2})$$

S_N hängt dabei vom Zufall ab, da er ja durch die Pfade des stochastischen Prozesses bestimmt wird. Für $N \to \infty$ sind zusätzlich Konvergenz- und Meßbarkeitsbetrachtungen nötig, die wir aber noch aufschieben wollen.

Die Wahrscheinlichkeitsgesetze, denen die auftretenden Gesamtschäden unterliegen, hängen bei unbeeinflußbaren p'_t, q'_t von der gewählten Betriebspolitik π' ab, da diese zur Definition des Maßes $P_{p'_o,\pi'}$ auf den meßbaren Pfadmengen geführt hat.

Es liegt nahe, wie in der statistischen Entscheidungstheorie die Erwartungswerte der Schäden in "geeigneter" Weise zur Beurteilung von Betriebspolitiken heranzuziehen. Zur präzisen Festlegung des Optimierungsziels haben wir im letzten Kapitel bereits eine Fülle von Optimalitätsbegriffen eingeführt.

Jetzt sei aber noch auf einige optimierende Modelle aus der Literatur eingegangen.

3.3 Optimierungsmodelle aus der Literatur

Während die Zahl der Arbeiten, die sich mit Warteschlangen- und Simulationsmodellen beschäftigen, sehr groß ist, findet man nur wenige, die das Problem der Optimierung behandeln. Olivier (1967) hat Rechenzeitzuteilungsverfahren zur Minimierung der mittleren Wartezeit untersucht. Unter der Voraussetzung, daß durch die Verwaltungsarbeit keine Verlustzeiten und keine Kosten entstehen und daß während der Abarbeitung der Aufträge keine neuen Aufträge eintreffen, gibt er notwendige Bedingungen für die Optimalität eines Zuteilungsverfahrens an. Marte (1970) und Küspert u. Marte (1970) knüpfen an die Resultate von Olivier an. Sie benutzen als Rechenzeitverteilungen stückweise exponentielle Verteilungen, die reale Rechenzeitverteilungen besser approximieren als die Exponentialverteilung, und die mit den Bedingungen von Olivier zu besonders einfachen Zuteilungsverfahren führen. Küspert u. Marte berücksichtigen zusätzlich noch Transportzeiten für das Hereinholen und Hinausschaffen der Aufträge.

Auch Charzan, Konheim u. Weiss (1968) und Konheim (1968) minimieren die mittlere Wartezeit. Auch bei ihnen dürfen während der Abarbeitung der Aufträge keine neuen eintreffen. Sie setzen außerdem diskrete Rechenzeitverteilungen voraus. Die Formulierung der Modelle und die Untersuchungsmethoden der bisher erwähnten Arbeiten zeigen keine Beziehung zur Theorie der stochastischen Entscheidungsmodelle. In Lehmann

(1966) und Lehmann (1968) wird anscheinend erstmalig die Anwendung endlicher, stationärer Markovscher Entscheidungsmodelle und die Benutzung allgemeiner Kosten zur Untersuchung des Zuteilungsproblems vorgeschlagen.
Aho, Denning u. Ullman (1971) benutzen ein nichtstationäres Entscheidungsmodell bei der Suche nach optimalen Seitenaustauschalgorithmen.
Die zuletzt genannten Modelle sind unmittelbar als Spezialfälle des hier betrachteten abstrakten Modells zu erkennen. Aber auch die übrigen lassen sich mit unseren Hilfsmitteln erfassen.
Die Heuristik bei der Formulierung unseres Modells zeigt eigentlich schon, daß es geeignet ist, eine große Klasse von Systemen leicht zu beschreiben. Die Zustände, Betriebsereignisse und Aktionen, die man für die Festlegung des Modells benötigt, kann man für ein gegebenes System erhalten, indem man dessen Arbeitsweise sorgfältig verfolgt und die relevanten Informationen in geeigneter Weise (z.B. numerisch verschlüsselt) in ein abstraktes Modell einbaut. Führt man das z.B. für das Modell von Konheim (1968) aus, so erhält man ein nichtstationäres Modell mit endlichen Mengen von Zuständen, Signalen und Aktionen. Als Optimierungsziel erhält man die Minimierung der Gesamtkosten in endlich vielen Schritten. Bei anderen Systemen, zum Beispiel bei Olivier (1967) und Küspert u. Marte (1970), können die Rechenzeiten beliebige positive, reelle Werte annehmen. Bei der Vergabe der Bedienungsstelle wird dort noch eine posi-

tive reelle Frist vorgegeben, nach der die Arbeit für diesen Benutzer spätestens unterbrochen wird, falls seine Wünsche nicht schon vorher voll befriedigt sind. Die Festlegung einer solchen Frist muß als eine Aktion in unserem Modell betrachtet werden. Hier erhalten wir also als Zustands-, Ereignis- und Aktionenmengen Mengen von der Mächtigkeit des Kontinuums. Um auch solche Modelle erfassen zu können, dürfen wir uns in unserem Modell nicht auf abzählbare oder gar nur endliche Mengen beschränken. Die allgemeineren Voraussetzungen erfordern allerdings eine Reihe maßtheoretischer Überlegungen, die bei abzählbaren Mengen trivial sind.

3.4 Bemerkungen zur praktischen Anwendbarkeit des Modells

Das Ziel bei der Konstruktion unseres Modells liegt zwar vorwiegend in dem Wunsch, das Verständnis der prinzipiellen Probleme zu fördern. Wir wollen aber doch kurz auf einige Schwierigkeiten eingehen, die bei dem Versuch auftreten, das Modell in der Praxis zu verwenden. Sie geben nämlich einige bemerkenswerte Hinweise auf die in der Wirklichkeit vorliegenden Verhältnisse.

Ein Teil der Schwierigkeiten ist von quantitativer Natur. Die Mengen der Zustände, Betriebsereignisse und möglichen Aktionen, die man zur vollständigen Beschreibung eines in der Praxis auftretenden Systems benötigt, sind in der Regel so groß, daß eine numerische Behandlung eines derartigen Modells selbst mit Hilfe großer Rechenanlagen auf ernste Kapazitätsprobleme stößt. Bereits die primitiven Modelle, die im 5. Kapitel betrachtet werden, zeigen diesen unangenehmen Effekt.
Diese Tatsache liefert eine starke Motivation für Versuche, die Modelle rein analytisch zu behandeln, um dadurch wenigstens Reduktionsmöglichkeiten zu finden, die es dann vielleicht gestatten, die Probleme so zu vereinfachen, daß sie numerisch behandelbar sind.

In unserem Modell wurde vorausgesetzt, daß die Wahrscheinlichkeitsverteilungen für die Betriebsereignisse und die Zustandsänderungen bekannt sind. Leider ist man in der Praxis von einer solchen Kenntnis noch weit entfernt. Während man

über das Verhalten der technischen Geräte relativ gute Aussagen gewinnen kann, ist das Problem beim Verhalten von Programmen und Benutzern an den Konsolen hinsichtlich ihrer Speicherzugriffe und Rechenzeitanforderungen erheblich schwieriger. Zwar liegen dazu eine Reihe von Meßergebnissen von laufenden Systemen vor (vgl.z.B. Coffman u. Wood 1966, Scherr 1966, 1967, Bryan 1967, Brawn u. Gustavson 1968 und Coffman u. Varian 1968), doch sind sie für unser Modell nicht detailliert genug. Eingehendere Messungen stoßen aber auf große praktische Schwierigkeiten. Um die relevanten statistischen Daten zu erhalten, müßte man in den Betriebsablauf eingreifen und eventuell das Betriebssystem zusätzlich mit umfangreichen Meßaufgaben belasten.

Aber selbst, wenn geeignete Meßverfahren zur Verfügung stehen, bleibt das Problem der Übertragbarkeit der Meßergebnisse von einer Anlage auf eine andere. Die Zusammensetzung des Benutzerkreises und die Art der vorwiegend behandelten Aufgaben kann doch sehr stark variieren, was sich naturgemäß auf das Verhalten der Programme auswirkt.
Ein Modell, das diesen Mangel an Kenntnissen voll ausdrücken soll, müßte adaptiv sein, das heißt die Möglichkeit enthalten, die während des Betriebsablaufs anfallenden Informationen für Rückschlüsse über die unbekannten Verteilungen ausnutzen und sich so immer besser an die bestehenden oder wechselnden Verhältnisse anpassen. Unser Modell ist nicht adaptiv. Für den Betrieb einer Rechenanlage ist uns auch kein adaptives Modell aus der Literatur bekannt.
Im Zusammenhang mit dem Verhalten der Benutzer ist noch ein

weiterer Gesichtspunkt wichtig, nämlich die Tendenz der einzelnen Benutzer, sich den Gegebenheiten an einem System anzupassen, um den Service für sich selbst zu verbessern. Wilkes (1970) berichtet vom CTSS, daß einmal insgeheim die Schwelle für die Programmlänge, oberhalb derer ein Benutzer benachteiligt wurde, von 10K auf 4K Worte herabgesetzt wurde. Obwohl diese Veränderung nicht bekannt gemacht wurde, hatte binnen weniger Wochen die mittlere Programmlänge um 40 % abgenommen. Coffman u. Kleinrock (1968) betrachten Zuteilungsalgorithmen und dazu Maßnahmen, die Benutzer ergreifen können, um den Service für sich zu verbessern. Aus dieser Sicht gleicht ein Rechenzentrum mit seinen Benutzern einem strategischen Spiel, bei dem jeder Mitspieler über einen Satz von Strategien verfügt und seinen eigenen Gewinn zu maximieren sucht, den er wiederum nach eigenen Maßstäben mißt. Der Spieler "Rechenzentrum" hat als Strategien dabei den Zuteilungsalgorithmus, disziplinarische Maßnahmen (Maschinensperre) und im kommerziellen Bereich die Preispolitik. Die Benutzer können z.B. ihre Programme in kleine Stücke zerlegen, wenn sie die Hoffnung haben, die einzelnen Teilstücke zusammen schneller gerechnet zu erhalten als das ungeteilte Programm; sie können eventuell zwischen verschiedenen Eingabewarteschlangen wählen u.ä.

Unser Modell betrachtet nicht die ganze Komplexität dieser Konkurrenzsituation. In unserem Modell wird vielmehr versucht, unter bestimmten Annahmen über die von den Benutzern verfolgten Verhaltensweisen die Politik des einen Spielers

"Rechenzentrum" zu optimieren. Die Möglichkeit, daß das "optimale" Betriebsprogramm eine Schwäche haben könnte, die eine radikale Änderung des Benutzerverhaltens bewirkt, so daß unter den neuen Verhältnissen eine völlig andere Betriebspolitik optimal ist, wollen wir hier nicht betrachten.

Eine weitere Schwierigkeit für die Anwendung des Modells liegt in der Notwendigkeit, die Schäden in den verschiedenen Situationen festzulegen. Es sollen doch so verschiedene Dinge wie die Verzögerung eines Benutzerprogramms, der Organisationsaufwand innerhalb der Anlage und mangelnde Auslastung von Maschinenkomponenten in einem Maßstab gegeneinander abgewogen werden. Bei der Bewertung der letzten beiden Einflüsse könnte man auf Überlegungen zurückgreifen, die zur Kostenerhebung für Dienstleistungen in Teilnehmerrechensystemen angestellt wurden (vgl.z.B. Rosenberg 1964, Selwyn 1970). Die Verzögerung von Benutzerprogrammen könnte man in einem derartigen System dadurch berücksichtigen, daß man mit den Benutzern Zahlungen vereinbart, die mit wachsender Wartezeit fallen (vgl. auch die Nutzenfunktionen bei Jessen 1968). Wir wollen hier auf diese Probleme nur hinweisen. Für unsere stochastischen Entscheidungsmodelle nehmen wir an, daß die Schäden in der in 3.2 beschriebenen Weise festliegen.

4. BETRIEBSMODELL UND STOCHASTISCHE ENTSCHEIDUNGSMODELLE

4.1 Reduktion des Betriebsmodells auf ein stochastisches Entscheidungsmodell

Dem von uns konstruierten stochastischen Modell für den Betriebsablauf in Teilnehmerrechensystemen können wir ein stochastisches Entscheidungsmodell zuordnen, um die Theorie dieser Modelle auf das Problem der Wahl einer optimalen Betriebspolitik anwenden zu können.

Wir kommen von unserem Modell zu einem stochastischen Entscheidungsmodell, indem wir einen Zustand des Betriebsmodells mit dem folgenden Betriebsereignis zu einem Zustand im Entscheidungsmodell zusammenfassen. Rein intuitiv liegt diese Zusammenfassung nahe; denn in einem Rechensystem stehen die zu einem Betriebsereignis gehörigen Informationen in gewissen Speicherzellen oder Registern, so daß man geneigt ist, zur Beschreibung des Zustandes des Gesamtsystems die Inhalte dieser Zellen hinzuzunehmen. Das Bewegungsgesetz des Entscheidungsmodells erhalten wir durch Zusammensetzen der Übergangswahrscheinlichkeiten q'_t und p'_t, die fest vorgegeben sind, zu der dann ebenfalls festen Übergangswahrscheinlichkeit q_t. An den Politiken ändert sich dadurch nichts, so daß für die späteren Überlegungen zur Optimierung nichts verloren geht. Wir setzen also:

$$(Z_t, \mathcal{Z}_t) := (S_t \times E_{t+1}, \mathcal{S}_t \times \mathcal{E}_{t+1}) \text{ für } t=0,1,2,\ldots$$
$$(A_t, \mathcal{A}_t) := (O_t, \mathcal{O}_t) \quad \text{für } t=1,2,3,\ldots$$

Die Mengen der Vorgeschichten H_t sind dann gleich V_{t+1}^{ω} mit den σ-Algebren $\mathcal{W}_{t+1}^{\omega}$ für t=0,1,2,...
Als Gewinnfunktion g_t setzen wir $-k_t$ für t=0,1,... und die Politiken π erhalten wir durch $\pi_t := \pi_t'$. Die Menge D der zulässigen Politiken ist dann gleich Π. Eine kleine Überlegung erfordert nur noch die Festlegung der Übergangswahrscheinlichkeiten q_t im Entscheidungsmodell und der Ausgangsverteilung p_o, falls eine Ausgangsverteilung für die Zustände im ursprünglichen Modell gegeben ist.
Die Frage nach der Ausgangsverteilung wird durch den Teil a) von Satz 1 in (2.3) erledigt. Wenden wir diesen Satz auf unsere Ausgangsverteilung p_o' und die Übergangswahrscheinlichkeit q_o' an, so erhalten wir die eindeutig bestimmte Ausgangsverteilung p_o auf $(Z_o, \mathcal{Z}_o)$, die wir dann im Entscheidungsmodell verwenden.
Zur Konstruktion von q_t (t=0,1,...) als Übergangswahrscheinlichkeit von $H_t \times A_{t+1} = V_{t+1}^{\sigma}$ nach $\mathcal{Z}_{t+1} = \mathcal{Y}_{t+1} \times \mathcal{E}_{t+2}$ sind als Bausteine die Übergangswahrscheinlichkeiten p_{t+1}' von V_{t+1}^{σ} nach $\mathcal{Y}_{t+1}$ und die Übergangswahrscheinlichkeit q_{t+1}' von $V_{t+1} \times S_{t+1} = V_{t+1}^{\varepsilon}$ nach $\mathcal{E}_{t+2}$ gegeben. Für festes $v \in V_{t+1}$ ist $p_{t+1}'(v;B)$ eine Wahrscheinlichkeit für B aus $\mathcal{Y}_{t+1}$ und $q_{t+1}'(v,s_{t+1};E)$ eine Übergangswahrscheinlichkeit von Z_{t+1} nach $\mathcal{E}_{t+2}$ $(E \in \mathcal{E}_{t+2})$.
Nach dem soeben zitierten Satz existiert daher für jedes v genau eine Wahrscheinlichkeit $q_t(v;.)$ auf $(S_{t+1} \times E_{t+2}, \mathcal{Y}_{t+1} \times \mathcal{E}_{t+2})$, für die gilt

$$q_t(v;B_1 \times B_2) = \int_{B_1} p_{t+1}'(v;ds_{t+1})\, q_{t+1}'(v,s_{t+1};B_2)$$

für $B_1 \in \mathcal{Y}_{t+1}, B_2 \in \mathcal{E}_{t+2}$.

Um nun q_t als Übergangswahrscheinlichkeit von $V^{\sigma}_{t+1} = H_t \times A_{t+1}$ nach $\mathcal{T}_{t+1} \times \mathcal{E}_{t+2}$ zu erkennen, muß man noch zeigen, daß $q_t(v;B)$ für festes $B \in \mathcal{T}_{t+1} \times \mathcal{E}_{t+2}$ $\mathcal{W}^{\sigma}_{t+1}$-meßbar ist.
Dazu benutzen wir den Teil b) von Satz 1 in (2.3). Diesen Satz wenden wir an mit $(E_1, \mathfrak{F}_1) = (V^{\sigma}_{t+1}, \mathcal{W}^{\sigma}_{t+1})$, $(E_2, \mathfrak{F}_2) = (S_{t+1}, \mathcal{T}_{t+1})$, $P^1_2 = p'_{t+1}$ und $X(v,s_{t+1}) = \chi_{B_1}(s_{t+1}) \cdot q'_{t+1}(v,s_{t+1};B_2)$ mit $B_1 \in \mathcal{T}_{t+1}$ und $B_2 \in \mathcal{E}_{t+2}$.
Dann folgt die $\mathcal{W}^{\sigma}_{t+1}$-Meßbarkeit von

$$q_t(v;B_1 \times B_2) = \int_{B_1} p'_{t+1}(v;ds_{t+1})\, q'_{t+1}(v,s_{t+1};B_2)$$

für jedes feste Rechteck $B_1 \times B_2$ aus $\mathcal{T}_{t+1} \times \mathcal{E}_{t+2}$, woraus auf dem üblichen Wege die Behauptung für alle $B \in \mathcal{T}_{t+1} \times \mathcal{E}_{t+2}$ folgt.
Wir können nun mit Hilfe von Satz 3 aus (2.3) aus den Bestimmungsstücken des stochastischen Entscheidungsmodells die eindeutig bestimmte Wahrscheinlichkeit $Q_{p_o\pi}$ auf

$$(S_o \times \prod_{\nu=1}^{\infty}{}^{*} (E_\nu \times O_\nu \times S_\nu),\ \mathcal{T}_o \times \bigotimes_{1}^{\infty} (\mathcal{E}_\nu \times \mathcal{O}_\nu \times \mathcal{T}_\nu))$$

konstruieren. Man sieht leicht ein, daß $Q_{p_o\pi}$ mit der in (3.1) konstruierten Wahrscheinlichkeit $P_{p_o\pi}$ übereinstimmt. Man braucht die Übereinstimmung nur auf den Rechteckzylindern mit endlicher Basis nachzurechnen und kann sich dann auf den Fortsetzungssatz berufen.
Insgesamt haben wir damit also unserem Modell ein stochastisches Entscheidungsmodell zugeordnet. Wir können daher die Begriffe aus Kapitel 2 benutzen.

Die folgende Tabelle stellt noch einmal die sich entsprechenden Stücke der beiden Modelle gegenüber:

Stochastisches Entscheidungsmodell	Betriebsmodell
$(Z_t, \mathcal{Z}_t)$	$(S_t \times E_{t+1}, \gamma_t \times \xi_{t+1})$: t=0,1,...
$(A_t, \mathcal{A}_t)$	$(O_t, \mathcal{O}_t)$: t=1,2,...
$(H_t, \mathcal{H}_t)$	$(V_t^{\omega}, \mathcal{W}_t^{\omega})$: t=0,1,2,...
p_o	$p_o' q_o'$
q_t	$p_t' q_t'$: t=0,1,2,...
π_t	π_t' : t=0,1,2,...
g_t	$-k_t$: t=0,1,2,...
D	$\mathcal{T}$

In den weiteren Abschnitten dieses Kapitels werden wir stochastische Entscheidungsmodelle etwas ausführlicher betrachten und einige für das Zuteilungsproblem interessante Sätze zitieren.

4.2 Existenzsätze und die Optimalitätsgleichung

Ohne zusätzliche Voraussetzungen über die Menge D der zulässigen Politiken in einem stochastischen Entscheidungsmodell garantieren die von uns bisher gemachten Voraussetzungen noch nicht einmal die Existenz von (γ, p, ε)-optimalen Politiken, wie ein einfaches Gegenbeispiel von Hinderer (1967) zeigt (vgl. a.a.O. p.32). Wie man an dem Gegenbeispiel weiter sieht, hilft nicht einmal die Voraussetzung, daß die Zustands- und Aktionenmengen endlich sind, denn in dem Gegenbeispiel treten nur zweipunktige Mengen auf. In den Untersuchungen über die Existenz optimaler Politiken spielen die Eigenschaften der Menge D der zulässigen Politiken eine entscheidende Rolle. Insbesondere benutzt man oft, daß gewisse Manipulationen an zulässigen Politiken wieder zulässige Politiken ergeben. Wir betrachten die Zusammensetzung gegebener Politiken zu einer neuen.
Sei also eine abzählbare Menge $\{\pi^{(\nu)}, \nu=1,2,\ldots\}$ von Politiken $\pi^{(\nu)} = (\pi_t^{(\nu)})$ gegeben. Die Menge H_t der Vorgeschichten zum Zeitpunkt t werde in eine abzählbare Menge von $\mathfrak{H}_t$-meßbaren, disjunkten Teilmengen $B_t^{(\nu)}$ zerlegt, so daß gilt

$$H_t = \sum_{\nu} B_t^{(\nu)} \quad \text{mit } B_t^{(\nu)} \in \mathfrak{H}_t .$$

Nun definiert man eine neue Politik $\hat{\pi} = (\hat{\pi}_t)$, indem man setzt

$$\hat{\pi}_t = \pi_t^{(\nu)} \text{ auf } B_t^{(\nu)} .$$

Die Politik heißt dann <u>erzeugt</u> von der Menge $\{\pi^{(\nu)}\}$. Eine Menge Π von Politiken heißt nun <u>σ-stabil</u> (resp. <u>stabil</u>), wenn

jede Politik $\hat{\pi}$, die von abzählbar (resp.endlich) vielen Politiken aus Π mit Hilfe einer geeigneten Zerlegung erzeugt wird, wieder zu Π gehört.
Jetzt gilt unter jeder der Voraussetzungen (EN), (EP) und (C) der folgende Satz von Hinderer (1967):

Satz 1:

Seien p ein Wahrscheinlichkeitsmaß auf $\mathcal{Z}_0$, $\varepsilon > 0, \gamma > 0$.

a) Ist D stabil, so existiert eine (γ,p,ε)-optimale Politik.

b) Ist D σ-stabil, so existiert eine (p,ε)-optimale Politik.

Ein Beispiel von Hinderer (1967) zeigt, daß die Stabilität einer Politikenmenge noch nicht die Existenz von (p,ε)-optimalen Politiken nach sich zieht. Ebenso ist σ-Stabilität nicht hinreichend für die Existenz von p-optimalen oder ε-optimalen Politiken, wofür Blackwell (1965) zwei Beispiele angibt.

Im Fall (C) läßt sich der angegebene Satz verschärfen (Hinderer 1967):

Satz 2:

Sei $\sum \|g_t\| < \infty$. Ist die Menge der zulässigen Politiken σ-stabil, so gibt es zu jeder Folge $p = (p_t)$ von Wahrscheinlichkeitsmaßen auf $\mathcal{Z}_t$ und zu jeder Folge $\varepsilon = (\varepsilon_t)$ positiver, reeller Zahlen eine stark (p,ε)-optimale Politik.

Eine wichtige Rolle als Optimalitätskriterium und auch als Hilfsmittel bei der expliziten Konstruktion optimaler Poli-

tiken spielt das von Bellman (vgl.z.B. Bellman 1957) formulierte Optimalitätsprinzip, das häufig in der Form einer Funktionalgleichung, der Optimalitätsgleichung, benutzt wird. Der Gültigkeitsbereich dieses Prinzips ist nicht genau bekannt und es wird häufig ohne Beweis benutzt, allein gestützt auf seine anschauliche Überzeugungskraft.
Dynkin (1965) und Hinderer (1967) haben die Optimalitätsgleichung für stochastische Entscheidungsmodelle unter verschiedenen Voraussetzungen bewiesen. Wir zitieren Hinderer (1967).

Satz 3:

$\{\Delta_t(h) \in \mathcal{U}_{t+1} : h \in H_t, t \in N\}$ sei eine Familie meßbarer Mengen mit der Eigenschaft

$$\pi_t(h, \Delta_t(h)) = 1 \text{ für alle } t \in N,\ h \in H_t \text{ und } \pi \in D.$$

D sei σ-stabil und es liege der konvergente Fall (C) vor. Weiter existiere zu jedem $\varepsilon > 0$ eine stark ε-optimale Politik. Dann gilt:

a) Der optimale Gewinn $G_t(h_t) = \sup_{\pi \in D} G_{t,\pi}(h_t)$ ist $\mathcal{H}_t$-meßbar.

b) Es ist

$$G_t(h) \leq \sup_{a \in \Delta_t(h)} \int \pi_t(h;da) \int q_t(h,a;ds)[g_t(h,a,s) + G_{t+1}(h,a,s)] = \sup_{a \in \Delta_t(h)} (L_{t,a}\, G_{t+1})(h)$$

für $t \in N$, $h \in H_t$.

c) Besteht D aus der Menge aller Politiken, für die $\pi_t(h, \Delta_t(h)) = 1$ für $t \in N$, $h \in H_t$ gilt, so genügen die optimalen Gewinne der Optimalitätsgleichung

$$G_t(h) = \sup_{a \in \Delta_t(h)} (L_{t,a}\, G_{t+1})(h) \text{ für } t \in N,\ h \in H_t.$$

Daß die Optimalitätsgleichung nicht immer erfüllt ist, sie in Anwendungsfällen also bewiesen werden muß, zeigt ein Beispiel von Hinderer (1967), in dem die optimalen Gewinne dieser Gleichung nicht genügen.

Die Optimalitätsgleichung liefert ein Kriterium für die starke Optimalität einer Politik.

Satz 4: Es liege der konvergente Fall (C) vor. Die Menge D der zulässigen Politiken erfülle die Voraussetzung von Satz 3,c. Dann ist eine Politik π aus D genau dann stark optimal, wenn die zugehörigen Gewinne $G_{t,\pi}$ der Optimalitätsgleichung genügen.

Voraussetzungen an die Mächtigkeiten der Aktionenmengen liefern weitere Existenzsätze.

Satz 5: Es liege wieder der konvergente Fall (C) vor und die Menge D der zulässigen Politiken sei σ-stabil.

a) Sind die Aktionenmengen A_t alle höchstens abzählbar, so gibt es zu jeder Folge $\varepsilon = (\varepsilon_t)$ positiver reeller Zahlen eine stark ε-optimale Politik. Die optimalen Gewinne bilden dann die einzige Lösung (u_t) der Optimalitätsgleichung mit

$$\|u_t\| \leq \sum \|g_t\| \quad , t \in N .$$

b) Sind die Aktionenmengen A_t alle endlich, so gibt es sogar eine stark optimale Politik.

Die hier und im folgenden zitierten Sätze findet man bei Hinderer (1967 oder 1970). In den meisten Fällen sind dort die Voraussetzungen etwas schwächer. Für unseren Überblick sind die schwächeren Voraussetzungen unnötig. Sie hätten die Darstellung nur undurchsichtiger gemacht.

4.3 Spezielle Entscheidungsmodelle und Politiken

Einige Entscheidungsmodelle und Politiken sind wegen ihrer einfachen Struktur von besonderer Bedeutung.

Eine Politik $\pi = (\pi_t)$ heißt Markovsch, falls die Übergangswahrscheinlichkeit $\pi_t(z_o, a_1, z_1, \ldots, a_t, z_t; A_{t+1})$ für jedes t und jedes A_{t+1} aus $\mathfrak{A}_{t+1}$ höchstens von den z_t aus Z_t abhängt.

Für ein Betriebsmodell einer Datenverarbeitungsanlage bedeutet das, daß die Entscheidungen des Organisationsprogramms sich nur auf den letzten Zustand und das zuletzt empfangene Signal stützen.

Sind die Meßräume $(Z_t, \mathfrak{Z}_t)$ und $(A_t, \mathfrak{A}_t)$ unabhängig von t, so heißt eine Markovsche Politik $\pi = (\pi_t)$ stationär, wenn π_t unabhängig von t ist.

Als Betriebspolitiken für Datenverarbeitungsanlagen sind natürlich die stationären Politiken besonders erstrebenswert, wenn nicht die einzigen, die praktisch sinnvoll sind. Weiter erscheinen Politiken, bei denen π_t eine beliebige Wahrscheinlichkeitsverteilung ist, von den Anwendungen her als unangenehm. Man wird es vorziehen wollen, die Entscheidungen deterministisch zu treffen. Eine Politik $\pi = (\pi_t)$ heißt deterministisch, wenn die Übergangswahrscheinlichkeit $\pi_t(h_t; .)$ für jedes t und jedes $h_t \in H_t$ in einem Punkt $a_{t+1}(h_t) \in A_{t+1}$ konzentriert ist. π_t läßt sich dann als $\mathfrak{H}_t$-meßbare Abbildung von H_t in A_{t+1} beschreiben (vgl.z.B. Neveu, 1969, p.95).

Analoge Begriffe hat man auch für die Entscheidungsmodelle eingeführt.

Ein stochastisches Entscheidungsmodell heißt Markovsch, wenn die Übergangswahrscheinlichkeit $q_t(z_o,a_1,..z_t,a_{t+1};C_{t+1})$ höchstens von (z_t,a_{t+1}) und g_t höchstens von (z_t,a_{t+1},z_{t+1}) abhängen, wobei $C_{t+1} \in \mathfrak{Z}_{t+1}$ für $t \in N$.

Ein Markovsches Entscheidungsmodell heißt stationär, wenn die folgenden Bedingungen erfüllt sind:

a) die Meßräume $(Z_t, \mathfrak{Z}_t)$ und $(A_t, \mathfrak{A}_t)$, sowie die Übergangswahrscheinlichkeiten q_t sind unabhängig von t;

b) die Gewinnfunktion g_t hat die Gestalt

$$g_t = \beta^t \cdot g,$$

wobei $g = g(z_t,a_{t+1},z_{t+1})$ und $0 < \beta \leq 1$, $t \in N$.

Sind das Entscheidungsmodell und die benutzte Politik beide Markovsch, so braucht der stochastische Prozeß $(\mathfrak{Z}_o,a_1,\mathfrak{Z}_1,...)$, der aus den Koordinatenvariablen des durch die Ausgangsverteilung p_o und die Politik π definierten Wahrscheinlichkeitsfeldes $(Z_o \times A_1 \times Z_1 \times .... \cdots \mathfrak{Z}_o \times \mathfrak{A}_1 \times \mathfrak{Z}_1 \quad ...., P_{p_o\pi})$ besteht, nicht unbedingt Markovsch zu sein, da die q_t ja von den Werten der beiden letzten Variablen abhängen können. Doch wird der stochastische Prozeß (σ_t) mit $\sigma_o = \mathfrak{Z}_o$, $\sigma_t = (a_t, \mathfrak{Z}_t)$ für t=1,2,... ein Markovscher Prozeß (vgl. Hinderer 1967,p.23).

Zuerst wollen wir uns der Frage zuwenden, ob es möglich ist, sich auf die Verwendung deterministischer Politiken zu beschränken, ohne beim Gewinn Einbußen hinnehmen zu müssen. Dazu bezeichne S_d die Menge aller deterministischen Politiken, S die Menge aller Politiken.

Satz 1 (Hinderer 1970, Strauch 1966):

Die Mengen Z_o, A_t, Z_t $(t=1,2,\ldots)$ seien Borelsche Mengen in vollständigen separablen metrischen Räumen. Die σ-Körper $\mathcal{Z}_t, \mathcal{A}_t$ seien die Borelschen Mengen in Z_t bzw. A_t.

Unter der Voraussetzung $\sum \|g_t^+\| < \infty$ gilt:

$$\sup_{\pi \in S_d} G_{t,\pi} = \sup_{\pi \in S} G_{t,\pi}$$

und

$$\sup_{\pi \in S_d} G_{\pi} = \sup_{\pi \in S} G_{\pi} .$$

Sind alle Politiken zulässig, so kann also im Falle (EN) der optimale Gewinn bereits mit deterministischen Politiken erreicht werden. Ob diese Aussagen auch z.B. im Falle (EP) gelten, ist nicht bekannt. Hinderer (1970) konnte sie nur unter der zusätzlichen Voraussetzung beweisen, daß alle g_t nach oben beschränkt sind.

Bei allgemeinen stochastischen Entscheidungsmodellen unterliegt die Menge D der zulässigen Politiken gewöhnlich noch Einschränkungen. Für die Existenz optimaler deterministischer Politiken sind daher noch Voraussetzungen über diese Menge zu machen. Außerdem sind in den bekannten Beweisen (vgl. Hinderer 1967, 1970) noch Annahmen über die Aktionenmengen

A_t notwendig. Wir zitieren drei Sätze von Hinderer (1967). Die Voraussetzungen sind dort allerdings etwas schwächer.

Satz 2: Die Mengen A_t, Z_t seien Borelsche Mengen in vollständigen, separablen metrischen Räumen. Die σ-Körper $\mathcal{Z}_t$, $\mathcal{A}_t$ seien die Borelschen Mengen in Z_t bzw. in A_t. $\sum g_t$ sei auf H gleichmäßig konvergent, $\sup \|g_t\| < \infty$ und

$$\sum \|g_t\| < \infty .$$

Weiter gebe es in A_{t+1} abgeschlossene Mengen $\Delta_t(h_t)$, so daß D die Menge aller Politiken $\pi = (\pi_t)$ ist, für die die Träger der Maße $\pi_t(h_t;.)$ in $\Delta_t(h_t)$ enthalten sind. Dann gibt es zu jeder Folge von Wahrscheinlichkeitsmaßen $p = (p_t)$ auf $\mathcal{Z}_t$ und zu jeder Folge $\varepsilon = (\varepsilon_t)$ positiver, reeller Zahlen eine stark (p,ε)-optimale deterministische Politik.

Zu stärkeren Aussagen kommt man durch Einschränkung der Mächtigkeit der Aktionenmengen:

Satz 3: Es gelte $\sum \|g_t\| < \infty$. D sei σ-stabil. Sind die Mengen A_t für t=1,2,... abzählbar und ist jede deterministische Politik zulässig, so gibt es zu jeder Folge $\varepsilon = (\varepsilon_t)$ positiver, reeller Zahlen eine stark ε-optimale deterministische Politik.

Satz 4: Es gelten die Voraussetzungen des vorhergenden Satzes. Die Mengen A_t seien alle endlich. Dann existiert eine stark optimale deterministische Politik.

Unter praktisch vernünftigen Voraussetzungen kann man sich also auf deterministische Politiken beschränken.

Die bisher betrachteten Politiken fällen ihre Entscheidungen zum Zeitpunkt t im allgemeinen unter Berücksichtigung der ganzen Vorgeschichte. Das Bewegungsgesetz wird aber häufig nur von einem Teil der Vorgeschichte abhängen. Eine speziell auch für Modelle von Zuteilungsalgorithmen für Datenverarbeitungsanlagen interessante Frage ist nun, ob man den maximalen Gewinn (falls überhaupt) nicht schon mit Politiken erzielen kann, die ebenfalls nur diesen Teil der Vorgeschichte berücksichtigen. Ein Beispiel von Blackwell (1965) (vgl. auch Strauch 1966) zeigt, daß das im allgemeinen nicht richtig ist. Dort werden ein Markovsches Entscheidungsmodell und eine Politik π' angegeben, zu der es keine Markovsche Politik π^* gibt mit

$$G_{o,\pi^*} \geq G_{o,\pi'} - \varepsilon \quad \text{für } \varepsilon < 1.$$

Mindestens bei der Suche nach optimalen und ε-optimalen Politiken hat man also einen Teil der weiteren Vorgeschichte zu berücksichtigen. Für allgemeine Modelle wird dieses Problem u.a. von Hinderer (1967,1970) näher untersucht. Wir wollen hier nur noch einen Satz von Strauch (1966) zitieren, der zeigt, daß man sich bei stationären Markovschen Entscheidungsmodellen, bei denen die Zustands- und Aktionenmengen Borelsche Teilmengen vollständiger, separabler metrischer Räume sind, auf Politiken beschränken kann, die außer dem letzten Zustand noch den Anfangszustand z_o mitberücksichtigen.

Satz 5: a) Es liege ein stationäres, Markovsches Entscheidungsmodell vor. Z und A seien Borelsche Teilmengen voll-

ständiger, separabler metrischer Räume. π sei eine Politik. Dann existiert eine im allgemeinen nicht deterministische Politik $\hat{\pi}$, bei der die bedingte Verteilung $\hat{\pi}_t$ auf $\mathcal{A}_t$ nur von z_n und z_o abhängt, so daß gilt:

$$G_{o,\pi}(z_o) = G_{o,\hat{\pi}}(z_o) \text{ für alle } z_o \in Z.$$

b) Ist p eine Wahrscheinlichkeit auf $(Z, \mathcal{Z})$, dann existiert eine Markovsche Politik π^{**}, so daß gilt:

$$\int G_{o,\pi} dp = \int G_{o,\pi^{**}} dp \ .$$

Aus dem zweiten Teil des Satzes können wir schließen, daß es eine M-p-optimale Politik gibt, die Markovsch ist, sobald es überhaupt eine M-p-optimale Politik gibt.

Bei der Betrachtung Markovscher Modelle wird man häufig zusätzlich annehmen können, daß die Ein-Schritt-Übergangswahrscheinlichkeiten nicht von der Zeit abhängen und das gleiche für die Erlöse gilt, daß das Modell also stationär ist. Es ist wichtig zu wissen, daß man sich dann in wichtigen Fällen auch auf stationäre Politiken beschränken kann. Dazu seien die folgenden Sätze zitiert, wobei die Zustands- und Aktionenmengen wieder Borelsche Mengen in vollständigen, separablen metrischen Räumen seien.

Satz 6 (Hinderer 1970):

Es sei der optimale Gewinn G größer als $-\infty$ und es gelte entweder "$g \leq o$" oder "g ist nach oben beschränkt und $\beta < 1$". Wenn es dann eine M-p-optimale Politik gibt, dann gibt es auch eine stationäre, deterministische M-p-optimale Politik.

Verschärft man in diesem Satz die Voraussetzungen dahin, daß man die Existenz einer optimalen Politik fordert, so folgt nach Blackwell (1965) und Strauch (1966), daß es auch eine stationäre, deterministische optimale Politik gibt.

Bisher haben wir uns im wesentlichen mit Existenzsätzen beschäftigt und die Frage nach der expliziten Konstruktion von Politiken, die in einem gewissen Sinne optimal sind, völlig außer Acht gelassen. Wir wollen uns jetzt diesem Problem zuwenden, aber auch hier nur einige wichtige Spezialfälle behandeln.

Falls wir den Gesamtgewinn nur für die ersten N Schritte maximieren wollen, setzen wir $g_t \equiv 0$ für $t > N$. Man spricht dann von einem Modell mit endlichem Planungshorizont. In diesem Fall gestattet es die Optimalitätsgleichung aus (4.2) Satz 3 häufig, im N-ten Schritt beginnend und zeitlich rückwärts fortschreitend eine optimale Politik explizit zu konstruieren (vgl.z.B. Howard (1960), Jewell (1963)).

Für den Fall eines unendlichen Planungshorizontes wollen wir uns jetzt auf stationäre Markoffsche Modelle beschränken, bei denen die Zustands- und Aktionenmengen endlich sind. Für diesen Fall hat Howard (1960,1965) einfache Konstruktionsverfahren angegeben.

Sei N die Anzahl der Zustände, n_i die Anzahl der zulässigen Aktionen im Zustand i, p_{ij}^k die Übergangswahrscheinlichkeit vom Zustand i in den Zustand j falls im Zustand i die Aktion k gewählt wurde. r_{ij}^k $(i,j=1,2,\ldots N,\ k=1,2,\ldots,n_i)$ sei der Erlös in einem Schritt, falls ein Übergang vom Zustand i in

den Zustand j bei der Aktion k erfolgte. Es sei

$$g_n(i,k,j) = \beta^{n-1} \cdot r_{ij}^k$$

mit $\beta < 1$. β kann als Diskontfaktor interpretiert werden. Howard hat gezeigt, daß dann der folgende Algorithmus eine optimale Politik liefert.

Es sei

$$q_i^k = \sum_{j=1}^{N} p_{ij}^k \, r_{ij}^k \quad (i=1,2,\ldots,N;\ k=1,2,\ldots,n_i).$$

Das Konstruktionsverfahren arbeitet iterativ, wobei jeder Iterationszyklus aus zwei Teilen besteht, einer "Wertbestimmung" und einer "Politikverbesserung".

Man kann nun von einer Politik ausgehen und zu dieser die Größen p_{ij}^k und q_i^k bestimmen, mit denen man die Wertbestimmung beginnt.

Wertbestimmung: Man bestimme die eindeutig bestimmte Lösung des Gleichungssystems

$$v_i = q_i^k + \beta \sum_{j=1}^{N} p_{ij}^k \, v_j \quad (i=1,2,\ldots,N).$$

Der Lösungsvektor $(v_1,v_2,\ldots,v_N)$ wird nun zur Verbesserung der Politik benutzt:

Politikverbesserung: Für jeden Zustand i suche man eine Aktion k_i', die die Prüfgröße

$$T_i(k) = q_i^k + \beta \sum_{j=1}^{N} p_{ij}^k \, v_j$$

mit den aus der Wertbestimmung erhaltenen v_j maximiert. Wird

das Maximum bereits durch die von der vorher in der Wertbestimmung benutzten Politik vorgeschriebenen Aktion erzielt, behalte man diese Aktion bei. Die neuen Aktionen bilden gemeinsam eine neue Politik. Ist diese neue Politik gleich der vorher benutzten Politik, so ist diese bereits die gesuchte optimale Politik. Andernfalls gehe man mit der neuen Politik wieder in die Wertbestimmung. Das Verfahren endet nach endlich vielen Schritten mit einer optimalen Politik.

Der Iterationszyklus kann auch mit der Politikverbesserung begonnen werden, wenn man zu Beginn $v_1 = v_2 = \ldots = v_N = 0$ setzt. Auch dann erhält man eine optimale Politik.

Im Falle $\beta = 1$ ist der Gesamtgewinn unendlich. Man kann dann den mittleren Gewinn pro Übergang maximieren. Howard (1960, 1965) hat für diesen Fall ein sehr ähnliches Iterationsverfahren angegeben. Die Situation ist aber jetzt komplizierter, da die Struktur der Markovschen Ketten, die nach der Wahl der Politiken durchlaufen werden, eine wesentliche Rolle spielt. Wir wollen hier nur den einfachsten Fall vorstellen und annehmen, daß man für jede Politik eine vollständig ergodische Markovsche Kette erhält. Dann sind die Zustandsgrenzwahrscheinlichkeiten $\bar{p}_1, \bar{p}_2, \ldots \bar{p}_N$ eindeutig bestimmt und der erwartete Gewinn pro Übergang ist

$$G = \sum_{i=1}^{N} \bar{p}_i \, q_i^k,$$

falls die q_i^k und die $\bar{p}_i$ zu einer festen Politik gehören.

Eine optimale stationäre Politik wird dann durch das folgende Iterationsverfahren erhalten:

Wertbestimmung: Man benutze die zur gegebenen Politik gehörigen p_{ij}^k, q_i^k (i,j=1,2,...,N) und bestimme die Lösung des linearen Gleichungssystems

$$g + v_i = q_i^k + \sum_{j=1}^{N} p_{ij}^k v_j \quad (i=1,2,\ldots,N)$$

$$v_N = 0.$$

g ist dann der Gewinn pro Übergang bei der gewählten Politik. Die sogenannten relativen Werte $v_1, v_2, \ldots, v_N$ werden zur Verbesserung der Politik benutzt.

Politikverbesserung: Man bestimme eine neue Politik, indem man für jedes i eine Aktion k_i' sucht, welche die Prüfgröße

$$T_i(k_i) = q_i^{k_i} + \sum_{j=1}^{N} p_{ij}^{k_i} v_j$$

mit den vorher bestimmten v_j maximiert. Wird das Maximum schon von einer Aktion erreicht, die von der vorher in der Wertbestimmung benutzten Politik vorgeschrieben wurde, so übernehme man diese Aktion ungeändert. Hat sich die alte Politik für kein i geändert, so ist bereits eine optimale Politik gefunden. Andernfalls gehe man mit der neuen Politik wieder in die Wertbestimmung ein. Das Verfahren liefert nach endlich vielen Schritten eine optimale Politik.

Auch hier kann man mit der Politikverbesserung beginnen, indem man zu Anfang $v_1 = v_2 = \ldots = v_N = 0$ setzt.

Die Howardschen Verfahren lassen sich für nicht zu große Zustands- und Aktionenmengen leicht für Rechenanlagen pro-

grammieren.

Für andere Konstruktionsverfahren sei auf die Arbeiten von Manne (1960), Jewell (1963), DeCani (1964), Derman (1966), Miller und Veinott (1969) und andere verwiesen.

5. EIN KONKRETES MODELL

5.1 Allgemeine Formulierung

In diesem Kapitel sollen die Erkenntnisse des vorhergehenden Kapitels auf ein Modell für ein Teilnehmerrechensystem angewendet werden, das etwas an ein Warteschlangenmodell von Krishnamoorthi und Wood 1966 erinnert.

Das System (vgl. Fig. 1) enthalte einen Rechnerkern (RK), einen Wecker (W) und n Bedienungsplätze B_i, an denen je maximal ein Auftrag zur Bearbeitung anstehen kann. Wir stellen uns vor, daß an einem Bedienungsplatz ein Benutzer eine Weile "denkt" und dann dem Rechner einen Auftrag erteilt. Nach einiger Zeit ist die Bearbeitung des Auftrags beendet, und der Benutzer überlegt sich aufgrund der Resultate einen neuen Auftrag. Die Abarbeitung der Aufträge werde von einem Zuteilungsprogramm (ZA) gesteuert. Die Anlage arbeite in Arbeitstakten gleicher Länge. Signale von den Bedienungsstationen, dem Wecker und dem Rechnerkern, sowie Aktionen des Zuteilungsprogramms können jeweils nur am Ende eines Arbeitstaktes wirksam werden. Diese Zeitpunkte seien mit t=0,1,2,... durchnumeriert.

Den Zustand der Anlage im Intervall (t,t+1) wollen wir wie folgt beschreiben (Änderungen seien nur zu den Zeitpunkten am Beginn und am Ende eines Intervalls möglich):

Der Rechnerkern kann während des Intervalls mit der Bear-

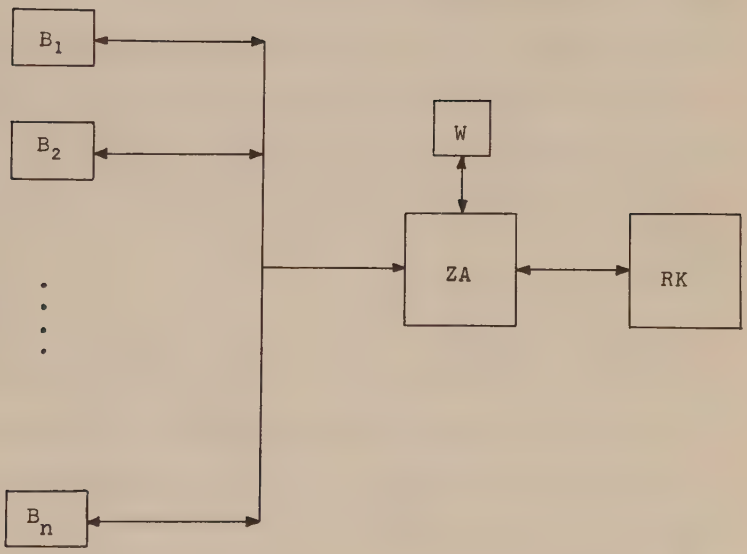

Fig. 1 Modell der Anlage

beitung eines Auftrags von einer Bedienungsstation B_i beschäftigt oder untätig sein. Eine Variable RW, die im ersten Fall den Wert i, im zweiten den Wert 0 annehme, beschreibe diese Tatsache. Wir haben also

(1) $\quad RW \in \{0,1,2,\ldots,n\} =: N_n \quad (n \geq 1).$

Der Wecker zeigt an, nach wie vielen Arbeitstakten ein Wecksignal eintreffen wird (der gerade laufende Takt werde hier und in analogen Fällen immer mitgezählt). Die Stellung des Weckers werde durch die Variable W beschrieben, für die gelte:

(2) $\quad W \in \{1,2,3,\ldots,\infty\} =: \mathbb{N}$.

Die Auftragssituation an einer Bedienungsstation B_i wollen wir durch mehrere Parameter beschreiben. Wir interessieren uns für die dort auftretenden Denkzeiten $d_i^{(j)}$ bis zum Eintreffen des j-ten Auftrags von B_i, die Anwesenheitszeit dieses Auftrags im System, die von ihm benötigte Rechenzeit $r_i^{(j)}$ und die Zeit, die er vom Zeitpunkt seiner Ankunft bis zum Beginn seiner Bearbeitung bzw. die er bei einer Unterbrechung zwischen zwei Bearbeitungsintervallen warten muß. Die Information über diese Größen für das Zuteilungsprogramm seien in den Zustandsvariablen DZ(i), AZ(i), RZ(i) und UZ(i) (i=1,....,n) auf folgende Weise enthalten:
Ist die Denkzeitvariable DZ(i) = k > 0, so bedeute das, daß an der Bedienungsstelle B_i seit k Arbeitsintervallen kein Auftrag ansteht, der Benutzer also bereits k Zeiteinheiten "denkt". DZ(i) sei gleich 0 genau dann, wenn von B_i noch ein

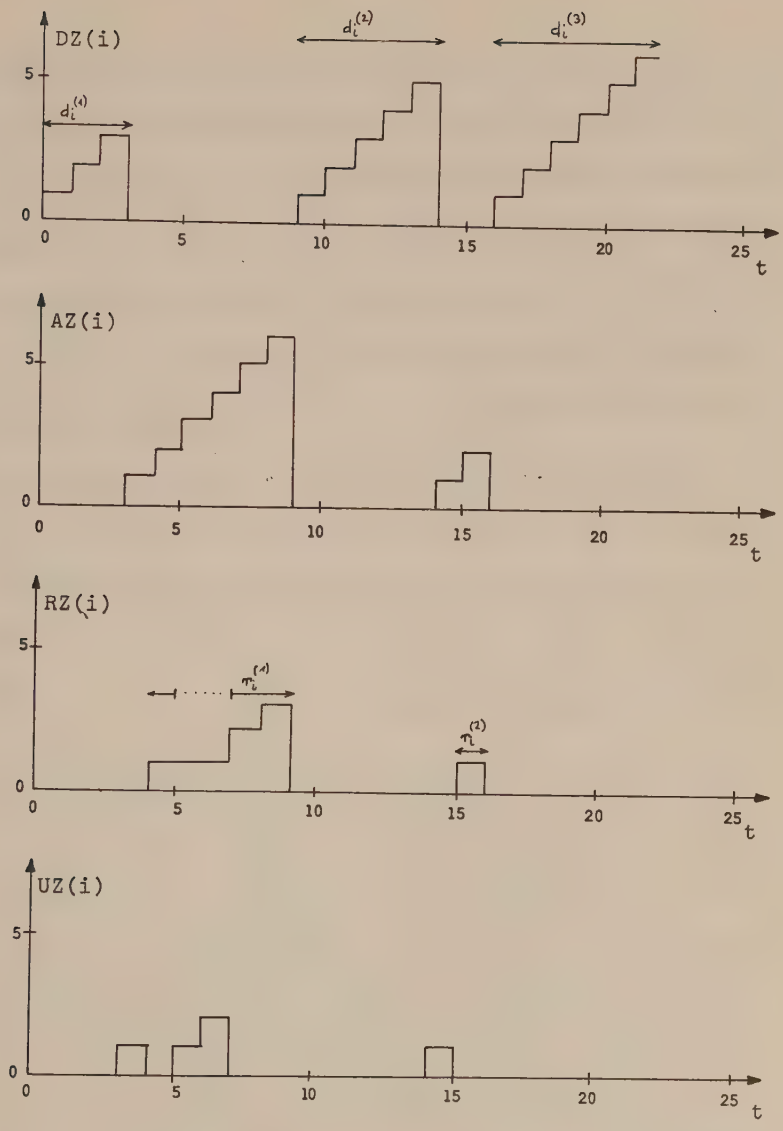

Fig. 2 Möglicher zeitlicher Verlauf der Zustandsvariablen an einer Bedienungsstelle

unvollendeter Auftrag vorliegt. Die Variable AZ(i) für die Anwesenheitszeit sei in einem bestimmten Intervall gleich 0 genau dann, wenn von der Bedienungsstation B_i kein Auftrag ansteht. Andernfalls gebe AZ(i) die Anzahl der Intervalle seit dem Eintreffen des Auftrags von B_i an. Die Rechenzeitvariable RZ(i) sei gleich 0, wenn von B_i kein Auftrag vorliegt. Sonst zähle sie die Intervalle, in denen der Auftrag bearbeitet wird. Die Unterbrechungszeitvariable UZ(i) sei nur dann von 0 verschieden, wenn von der Bedienungsstelle B_i ein Auftrag gestellt ist, dieser aber nicht bearbeitet wird. Ist der Auftrag seit seinem Eintreffen noch gar nicht bearbeitet worden, so zählt UZ(i) die Arbeitstakte seit seinem Eintreffen, sonst die Arbeitstakte seit der letzten Unterbrechung der Bearbeitung.
Für diese Parameter gilt also:

$$
\begin{aligned}
&DZ(i) \in \{0,1,2,....\} =: N_0,\\
&AZ(i) \in \{0,1,2,....\} = N_0,\\
&RZ(i) \in \{0,1,2,....\} = N_0,\\
&UZ(i) \in \{0,1,2,....\} = N_0.
\end{aligned}
\tag{3}
$$

Zur Illustration stellt Figur 2 einen möglichen zeitlichen Verlauf dieser Zustandsvariablen an einer Bedienungsstation dar: In den ersten drei Zeitintervallen "denkt" der Benutzer und erteilt dann einen Auftrag, der aber nicht sofort bearbeitet wird (UZ(i) = 1). Erst im fünften Arbeitsintervall beginnt die Bearbeitung. Sie wird aber nach einem Arbeitstakt wieder für zwei Takte unterbrochen. Nach zwei weiteren Takten ist der Auftrag dann erledigt. In den fünf folgenden

Takten bereitet der Benutzer einen neuen Auftrag vor.

Nach der Interpretation, die wir den Parametern zugewiesen haben, ist es klar, daß sie ihre Werte nicht unabhängig voneinander aus den in (3) beschriebenen Mengen annehmen können. So schließt zum Beispiel DZ(i) > 0 positive Werte der anderen Parameter an der Bedienungsstation B_i aus. Die Menge der real zulässigen Zustände zu einem festen Zeitpunkt an der Bedienungsstation B_i ist also eine echte Teilmenge von N_0^4, der Menge der formal zulässigen Zustände. Sei nun Z die Menge der real zulässigen Zustände der gesamten Anlage. Für einen Anlagenzustand z gilt dann:

z=(W,RW,DZ(1),AZ(1),RZ(1),UZ(1),...,DZ(n),AZ(n),RZ(n),UZ(n))

mit

(4) $$z \in Z \subset \mathbb{N} \times N_n \times (N_0^4)^n.$$

Der Zuteilungsalgorithmus (ZA) wird jeweils am Ende eines Arbeitstaktes wirksam. Für ihn sind Signale vom Rechnerkern, vom Wecker und von den Bedienungsplätzen relevant. Der Bedienungsplatz B_i kann im Laufe eines Intervalls die Ankunft eines neuen Auftrags melden. In unserem Modell werde das dadurch beschrieben, daß eine Signalvariable AS(i)=1 gesetzt wird; bleibt eine solche Meldung während eines Intervalls aus, so sei AS(i)=0 für dieses Intervall.
Vom Rechnerkern könnte gemeldet werden, daß der gerade bearbeitete Auftrag beendet ist und das System verläßt. Eine entsprechende Variable ES sei in diesem Fall gleich 1, sonst 0. Der Wecker kann signalisieren, daß ein voreingestelltes

Zeitintervall, das aus mehreren Arbeitstakten bestehen kann, abgelaufen ist. Diese Meldung werde mit Hilfe der Variablen US durch US=1 beschrieben. US=0 kennzeichne das Ausbleiben des Weckersignals.

Derartige Signale können natürlich in gewissem Umfang gleichzeitig eintreffen. Unter einem Betriebssignal e verstehen wir daher:

(5) $\quad e = (US,ES,AS(1),\ldots,AS(n)) \in \{0,1\}^{n+2}$.

E sei die Menge der Betriebssignale.

Der Zuteilungsalgorithmus hat im Prinzip die Aufgabe, einen der anstehenden Aufträge auszuwählen und den Wecker auf eine gewisse Frist einzustellen, nach deren Ablauf die Bearbeitung des ausgewählten Auftrags spätestens unterbrochen wird, um eine neue Rechnerkernvergabe zu ermöglichen. Eine solche Aktion des Zuteilungsalgorithmus kann man durch ein Paar (k,Q) beschreiben, wobei k=i bedeute, daß ein Auftrag von der Bedienungsstelle B_i zur Bearbeitung durch den Rechnerkern ausgewählt wird (k=0, wenn der Rechnerkern untätig ist). Q = 1,2,... bedeute, daß der Wecker nach 1,2,... Arbeitstakten (Quanten) sein Signal gibt. Für eine Aktion a aus der Menge A aller Aktionen gilt also

(6) $\quad a = (k,Q) \in A \subset N_n \times \mathbb{N},$

wobei die Mengen N_n und $\mathbb{N}$ in (1) bzw. (2) definiert wurden. Q = ∞ bedeute, daß der betreffende Auftrag auf jeden Fall zu Ende gerechnet wird.

Die Mengen Z und A sind abzählbar, E ist endlich. Wir ver-

sehen diese Mengen mit ihren Potenzmengen als σ-Algebren. Maßtheoretische Probleme tauchen in diesem Modell also nicht auf.

Die Deutung von Q als Zeitvorgabe schließt es aus, daß die Bearbeitung eines Auftrags, dem diese Vorgabe gewährt worden ist, vor Ablauf dieser Frist unterbrochen wird. Es sei denn, durch ein Signal vom Rechnerkern erfolgt eine Endemeldung, die die vollständige Erledigung des Auftrags vor Ablauf der zugestandenen Frist anzeigt. Das gibt für die Menge der zulässigen Politiken $\pi = (\pi_1, \pi_2, \ldots)$ die folgende Einschränkung: $\pi_t(z_o, e_1, a_1, z_1, \ldots, a_{t-1}, z_{t-1}, e_t; B)$ ist als Funktion von $B \in \mathcal{A}_t$ für $z_i \in Z$ $(i=0,1,\ldots t-1)$, $a_i \in A$ und $e_i \in E$ $(i=1,2,3,\ldots,t)$, mit $a_{t-1} = (k_{t-1}, Q_{t-1})$, $z_{t-1} =$

$= (W_{t-1}, RW_{t-1}, DZ(1)_{t-1}, \ldots, DZ(n)_{t-1}, AZ(n)_{t-1}, RZ(n)_{t-1}, UZ(n)_{t-1})$

und $e_t = (US_t, ES_t, AS(1)_t, \ldots, AS(n)_t)$ eine beliebige Wahrscheinlichkeit auf $\mathcal{A}_t$, wenn $Q_{t-1} = 1$, die Zeitvorgabe also erschöpft ist, oder $ES_t = 1$, der gerade bearbeitete Auftrag also erledigt ist, oder $RW_{t-1} = i \neq 0$ und $DZ(i)_{t-1} > 0$ ist, wenn der Rechnerkern einem freien Bedienungsplatz zugeordnet ist. In den übrigen Fällen muß diese Wahrscheinlichkeit im Punkt (k_t, Q_t) konzentriert sein mit $k_t = k_{t-1}$ und $Q_t = Q_{t-1} - 1$. Es muß nämlich derselbe Auftrag wie im vorhergehenden Intervall bearbeitet werden und die Anzahl der Intervalle, die noch bis zur nächsten Meldung vom Wecker verbleiben, ist um 1 geringer geworden. Die in (4.2) Satz 3 eingeführten Mengen $\Delta_t(h)$ der zulässigen Aktionen bei Vorliegen der Vorgeschichte h sind demnach

(7)
$$\Delta_t(z_o,e_1,a_1,z_1,\ldots,a_{t-1},z_{t-1},e_t) = \begin{cases} A, \text{ falls } Q_{t-1}=1 \text{ oder } ES_t=1 \\ \quad \text{oder } RW_{t-1}=i\neq 0 \text{ und } DZ(i)_{t-1}>0 \\ (k_{t-1},Q_{t-1}-1) \quad \text{sonst.} \end{cases}$$

Die Menge D aller zulässigen Politiken $\pi = (\pi_1,\pi_2,\ldots)$ bestehe aus allen Politiken mit

(8) $\pi_t(h,\Delta_t(h)) = 1$ für $t=1,2,\ldots,h=(z_o,\ldots,a_{t-1},z_{t-1},e_t)$.

Man sieht unmittelbar ein, daß die Menge D der zulässigen Politiken σ-stabil ist.

Als nächstes betrachten wir die Wahrscheinlichkeitsgesetze, nach denen die Betriebssignale eintreffen. Das Weckersignal trifft nach unseren Konventionen im Intervall (t,t+1) genau dann ein, wenn $W_t=1$ ist. Es gilt also:

(9) $$P(US_{t+1}=1) = 1 - P(US_{t+1}=0) = \begin{cases} 1, & \text{falls } W_t = 1 \\ 0 & \text{sonst.} \end{cases}$$

Mit Hilfe des Kroneckersymbols können wir die Wahrscheinlichkeit $PU(W_t; US_{t+1})$ dafür, daß die Weckervariable den Wert US_{t+1} annimmt, falls die Weckerzustandsvariable den Wert W_t hat, einfach darstellen:

(10) $$PU(W_t; US_{t+1}) = \delta_{1,US_{t+1}} \cdot \delta_{1,W_t} + (1-\delta_{1,US_{t+1}})(1-\delta_{1,W_t}).$$

Die Wahrscheinlichkeit, daß an der Bedienungsstation B_i im Intervall (t,t+1) ein neuer Auftrag eintrifft, ist für DZ(i)=0 jedenfalls gleich 0, da in diesem Fall nach unseren Konventionen an dieser Stelle noch ein unvollendeter Auftrag ansteht und jeweils maximal nur ein Auftrag zugelassen ist.

Unter der Denkzeit $d_i^{(j)}$ am Bedienungsplatz B_i wollen wir die Zeit verstehen, die vom Start des Systems bis zum Eintreffen des ersten Auftrags an dieser Station vergeht bei $j=1$, bzw. die Zeit zwischen der Vollendung des $(j-1)$-ten Auftrags und dem Eintreffen des nächsten an derselben Bedienungsstelle für $j \geq 2$. Die im allgemeinen verschiedenen an einer Bedienungsstelle auftretenden Denkzeiten seien unabhängige, identisch verteilte Zufallsgrößen auf einem Wahrscheinlichkeitsfeld $(M, \mathfrak{K}, P)$, die nur positive ganze Zahlen annehmen. $P^i(n)$ sei die Wahrscheinlichkeit, daß die Denkzeit d_i den Wert n annimmt, also genau n Arbeitstakte umfaßt.

Hat nun in unserem Modell der Denkzeitparameter den Wert DZ(i), so ist die Wahrscheinlichkeit dafür, daß genau im aktuellen Intervall an der Bedienungsstelle B_i ein Auftrag eintrifft

$$(11)\quad P_i(DZ(i)) := P(d_i = DZ(i) \mid d_i \geq DZ(i))$$

$$= \frac{P^i(DZ(i))}{P(d_i \geq DZ(i))}\ , \text{ falls } P(d_i \geq DZ(i)) > 0\ .$$

Falls die im Nenner auftretende Wahrscheinlichkeit verschwindet, kann die linke Seite beliebig festgelegt werden. Wir setzen sie dann mit 0 fest und im Einklang hiermit $P_i(0)=0$.

So geben die $P_i(DZ(i))$ für $i=1,2,\ldots n$ genau die Wahrscheinlichkeit für das Eintreffen eines neuen Auftrags an der Bedienungsstation B_i an, falls bereits DZ(i) Zeitintervalle seit dem Ausscheiden des letzten Auftrags vergangen sind.

Wir können für diese Ankunftswahrscheinlichkeit PA_i jetzt schreiben:

(12) $PA_i(DZ(i);AS(i)=k)=P_i(DZ(i)).\delta_{k,1}+(1-P_i(DZ(i))\cdot(1-\delta_{k,1})$

für k=0,1 .

Die Verteilungen an den verschiedenen Bedienungsstationen wollen wir als unabhängig voraussetzen. Sie sollen weiter vom Weckersignal und von den jetzt zu besprechenden Endesignalverteilungen unabhängig sein.

Von den Rechenzeiten $r_i^{(j)}$ der Aufträge von einer Bedienungsstation B_i werde vorausgesetzt, daß sie unabhängige, identisch verteilte Zufallsgrößen zum Wahrscheinlichkeitsfeld $(M, \mathfrak{A}, P)$ sind, die als Wertebereich die natürlichen Zahlen besitzen. $Q^i(n)$ sei die Wahrscheinlichkeit, daß die Rechenzeit $r_i^{(j)}$ genau n Arbeitstakte umfaßt.

Die Wahrscheinlichkeit, daß ein Auftrag von der Station B_i, der bereits RZ(i) Arbeitstakte bedient worden ist (den gerade laufenden Takt mitgerechnet), genau in diesem Takt beendet wird, ist also

(13) $Q_i(RZ(i)) = P(r_i=RZ(i) \mid r_i \geq RZ(i))$

$$= \frac{Q^i(RZ(i))}{P(r_i \geq RZ(i))} \text{ , falls } P(r_i \geq RZ(i)) > 0.$$

Wir setzen noch $Q_i(0) = 0$ und $Q_i(RZ(i)) = 0$, falls der Nenner in (13) verschwindet.

Die Wahrscheinlichkeit PE für das Eintreffen des Endesignals hängt nur davon ab, welcher Auftrag bearbeitet wird und wie

viele Arbeitstakte der betreffende Auftrag bereits bedient worden ist. Wir können schreiben:

$$(14)\quad PE(RW,RZ(1),RZ(2),\ldots,RZ(n);ES) =$$

$$(1-\delta_{o,RW})\left[Q_{RW}(RZ(RW))\cdot\delta_{ES,1} + (1-Q_{RW}(RZ(RW)))\cdot\delta_{ES,o}\right] +$$

$$+ \delta_{o,RW}\cdot\delta_{ES,o}\ .$$

Sei nun der Zustand zum Zeitpunkt t

$$z_t=(W_t,RW_t,DZ(1)_t,\ldots,DZ(n)_t,AZ(n)_t,RZ(n)_t,UZ(n)_t).$$

Dann folgt aus (10),(12),(14) und der Unabhängigkeitsvoraussetzung für die Wahrscheinlichkeit eines ankommenden Signals:

$$(15)\quad q_{t+1}(z_t;(US,ES,AS(1),\ldots,AS(n))) =$$

$$= PU(W_t,US)\cdot PE(RW_t,RZ(1),\ldots,RZ(n)_t,ES)\prod_{i=1}^{n} PA_i(DZ(i),AS(i)).$$

Die Wahrscheinlichkeit q_{t+1} hängt nur vom letzten Zustand z_t ab.

Nun müssen wir noch die Wirkung der Aktionen beschreiben. Aus der Interpretation der angegebenen Parameter folgern wir:

Seien z_t = (W,RW,DZ(1),...,DZ(n),AZ(n),RZ(n),UZ(n)),

e_{t+1} = (US,ES,AS(1),AS(2),...,AS(n)),

a_{t+1} = (k,Q) und

z_{t+1} = (W',RW',DZ(1)',...,DZ(n)',AZ(n)',RZ(n)',UZ(n)').

Dann gilt

W' = Q

RW' = k

$$(16)\ DZ(i)' = \begin{cases} 0, & \text{falls } (DZ(i)=0 \wedge RW \neq i) \vee (DZ(i)=0 \wedge RW=i \wedge ES=0) \vee \\ & \quad \vee (DZ(i)>0 \wedge AS(i)=1) \\ 1, & \text{falls } DZ(i)=0 \wedge ES=1 \wedge RW=i \\ DZ(i)+1, & \text{falls } DZ(i)>0 \wedge AS(i)=0 \end{cases}$$

$$(17)\ AZ(i)' = \begin{cases} 0, & \text{falls } (AZ(i)=0 \wedge AS(i)=0) \vee (AZ(i)>0 \wedge RW=i \wedge ES=1) \\ 1, & \text{falls } AZ(i)=0 \wedge AS(i)=1 \\ AZ(i)+1, & \text{falls } (AZ(i)>0 \wedge RW \neq i) \vee (AZ(i)>0 \wedge RW=i \wedge ES=0) \end{cases}$$

$$(18)\ RZ(i)' = \begin{cases} 0, & \text{falls } AZ(i)'=0 \\ RZ(i), & \text{falls } k \neq i \wedge AZ(i)'>0 \\ RZ(i)+1, & \text{falls } k=i \wedge AZ(i)'>0 \end{cases}$$

$$(19)\ UZ(i)' = \begin{cases} 0, & \text{falls } AZ(i)'=0 \vee RW'=i \\ UZ(i)+1, & \text{falls } AZ(i)'>0 \wedge RW' \neq i. \end{cases}$$

Endlich müssen wir noch die Kostenfunktionen festlegen. Mit dem Eintreffen des Weckersignals sind besondere Maßnahmen erforderlich, die einen zusätzlichen Aufwand erfordern (z.B. die Feststellung, welcher Auftrag als nächster bearbeitet werden soll). Dieser Aufwand werde durch die Kosten KN beschrieben, die also genau dann eintreten, wenn das Weckersignal eintrifft. Allgemein betragen daher die Weckerkosten KW in einem Intervall mit dem Weckersignal US

$$(20) \qquad KW = KN \cdot \delta_{US,1} \cdot$$

Es ist erwünscht, daß die Aufträge an ein Rechensystem so schnell wie möglich erledigt werden. Dazu kann man versuchen, deren Anwesenheitszeit im System zu minimieren. Die Anwesenheitszeit eines Auftrags von der Bedienungskonsole B_i werde

also mit den Kosten KOSA(i,AZ(i)) bewertet, wenn er das AZ(i)-te Intervall ansteht. KOSA(i,0) = 0 ist dann klar.

Die Kosten KOSA bewerten auch die Intervalle negativ, in denen der Auftrag bearbeitet wird. Meist ist es jedoch sinnvoller, nur die Verzögerungen negativ zu bewerten, also nur die Zeit, die ein Auftrag länger im System verbleibt als es seiner verbrauchten Rechenzeit nach nötig gewesen wäre. Diese Verzögerungen werden durch Verzögerungskosten KOSV(i,VZ(i)) bewertet, die bei einem Auftrag B_i anfallen, wenn er das VZ(i)-te Arbeitsintervall im System ansteht ohne bearbeitet zu werden. Wir setzen noch KOSV(i,0) = 0. Die Verzögerungskosten in einem Intervall für einen Auftrag von B_i, der in diesem Intervall nicht bearbeitet wird, der aber bereits AZ(i) Intervalle im System anwesend ist und RZ(i) Intervalle bearbeitet wurde, betragen also KOSV(i,AZ(i)-RZ(i)).

In anderen Fällen mag es wichtig sein, die Länge der Unterbrechungsintervalle für einen Auftrag klein zu halten. Diesem Zweck dienen die Kosten KOSW(i,UZ(i)), die in einem Intervall für den Auftrag von B_i entstehen, wenn seit seinem Eintreffen oder seit seiner letzten Unterbrechung UZ(i) Zeitintervalle vergangen sind. Wir setzen wieder KOSW(i,0)=0.

Schließlich erfordert die Unterbrechung eines Auftrags einen organisatorischen Aufwand, da im allgemeinen u.a. Speicherbereiche geräumt und neu beschrieben werden müssen. Der Umfang dieser Tätigkeiten kann davon abhängen, wie weit der unterbrochene Auftrag bereits bearbeitet wurde. Die Unterbrechungskosten KOSU(i,RZ(i)) hängen daher für einen Auftrag

von B_i von seiner verbrauchten Rechenzeit ab. Sie entstehen genau dann, wenn im Zustand z_t $RW_t = i$ ist, aber in z_{t+1} gilt $RW_{t+1} \neq i$ und $AZ(i) > 0$.

Bei dieser Festlegung der Kosten hängen die Gesamtkosten im Zeitintervall (t,t+1), die sich additiv aus den aufgeführten Einzelkosten ergeben, nur von z_t und z_{t+1} ab. Wir setzen noch voraus, daß alle Kosten durch eine Konstante C beschränkt sind.

Schließlich werde noch ein Diskontfaktor $\beta \leq 1$ benutzt.

Auf dieses Modell wollen wir die Erkenntnisse des vorigen Kapitels anwenden.

5.2 Einige Ergebnisse

Die Zustands- und Aktionenmengen des in (5.1) beschriebenen Modells sind alle abzählbar. Setzen wir voraus, daß $\beta < 1$ ist, dann gilt

$$(1) \qquad \sum \|g_t\| = \sum \beta^t \cdot \|g\| \leq C \sum \beta^t = \frac{C}{1-\beta} < \infty \quad .$$

Es liegt also der konvergente Fall (C) vor.
Daß die Menge D der zulässigen Politiken σ-stabil ist und alle möglichen deterministischen Politiken enthält, ist im vorigen Abschnitt bereits festgestellt worden.
Satz 1 aus (4.2) liefert uns also sofort die Existenz (p,ε)-optimaler Politiken. Da wir sogar den konvergenten Fall haben, können wir auch Satz 2 anwenden, der uns die Existenz stark (p,ε)-optimaler Politiken liefert. Die Abzählbarkeit der Aktionenmengen garantiert uns darüber hinaus nach Satz 5 (4.2) die Existenz stark ε-optimaler Politiken. Wir können daher auch Satz 3 dieses Abschnitts anwenden und wissen daher, daß die optimalen Gewinne der Optimalitätsgleichung genügen und daß eine Politik π genau dann stark optimal ist, wenn die zu ihr gehörigen Gewinne der Optimalitätsgleichung genügen (Satz 4, 4.2).
Die Existenz einer stark optimalen Politik konnte allerdings weder gezeigt noch widerlegt werden. Diese Frage bleibt also offen.
Aus (4.3), Satz 3 können wir weiter schließen, daß eine stark ε-optimale Politik bereits in der Menge der determi-

nistischen Politiken gefunden werden kann.

Die Meßräume $((Z_t,E_{t+1}),\ \mathfrak{Z}_t \otimes \mathfrak{E}_{t+1})$ $(t=0,1,2,\ldots)$ und $(A_t, \mathfrak{A}_t)$ für $t=1,2,\ldots$ hängen in unserem Modell nicht von t ab. Die Übergangswahrscheinlichkeiten q_{t+1} nach (Z_t,E_{t+1}) hängen nur von z_{t-1}, e_t und a_t aber nicht von t ab. Die Gewinne g_t hängen nur von z_t, z_{t+1} und von t über den Diskontfaktor β ab:

$$g_t = \beta^{t-1} \cdot g \tag{2}$$

Es liegt also ein stationäres Markovsches Entscheidungsmodell vor. Die Voraussetzungen von Satz 6 aus (4.3) sind erfüllt. Wir wissen also aus der Bemerkung nach diesem Satz, daß es eine stationäre, deterministische optimale Politik gibt, falls es überhaupt eine optimale Politik gibt.

In einem wirklichen Teilnehmersystem wird man unendliche Zeitvorgaben Q nicht in Betracht ziehen. Man liefe dann Gefahr, daß zum Beispiel durch eine irrtümlich programmierte unendliche Schleife das System völlig blockiert wird. Man hat daher eine Schranke WMAX, die die maximal zulässige Zeitvorgabe festlegt. Die dann noch zulässige Menge von Aktionen ist endlich und es gilt für jede Aktion a

$$a = (k,Q) \in N_n \times \{1,2,\ldots,WMAX\}. \tag{3}$$

Jetzt ist der Teil b) von Satz 5 in (4.2) anwendbar, der uns die Existenz einer stark optimalen Politik garantiert. Da eine stark optimale Politik erst recht optimal ist, folgt aus Satz 6 in (4.3) die Existenz einer stationären, deterministischen optimalen Politik.

Diese können wir nach dem Verfahren von Howard (1960) für den diskontierten Fall in der in (4.3) beschriebenen Weise berechnen, falls auch die Zustandsmengen endlich sind.

5.3 Endliche Zustandsmengen

Durch die Beschränkung der Zeitvorgabe Q auf maximal WMAX haben wir nicht nur erreicht, daß die Aktionenmengen endlich sind, sondern auch, daß der Parameter W, der die Weckerstellung beschreibt, maximal diesen Wert annehmen kann. Wir haben also:

(1) $$W \in \{1,2,\ldots,WMAX\} =: \bar{N}_{WMAX} .$$

Bei den Parametern zur Charakterisierung der Auftragssituation an der Bedienungskonsole B_i wird man in der Praxis auch nur endlich viele Werte zulassen können. Dazu wird man für jeden Parameter einen Maximalwert vorschreiben (möglicherweise verschiedene Maximalwerte für verschiedene Bedienungsstellen). Wenn diese Maximalwerte überschritten sind, wird dann nur noch registriert, daß jeweils mindestens diese Anzahl von Intervallen vergangen ist. Um dieser Situation gerecht zu werden, müssen wir an dem Modell in (5.1) einige Änderungen vornehmen.

Die Maximalwerte für die Denkzeitparamter DZ(i), die Anwesenheitsparamter AZ(i), die Unterbrechungszeitparameter UZ(i) und die Rechenzeitparameter RZ(i) seien DZM(i), AZM(i), UZM(i) und RZM(i) (i=1,2,...,n).

Die Bedingungen (3) aus (5.1) nehmen dann die Gestalt an:

(2)
$$\begin{aligned} DZ(i) &\in \{0,1,2,\ldots,DZM(i)\} =: N_{DZM(i)} \\ AZ(i) &\in \{0,1,2,\ldots,AZM(i)\} =: N_{AZM(i)} \\ RZ(i) &\in \{0,1,2,\ldots,RZM(i)\} =: N_{RZM(i)} \\ UZ(i) &\in \{0,1,2,\ldots,UZM(i)\} =: N_{UZM(i)} \end{aligned}$$

für i=1,2,...,n.

Für jeden Zustand z=(W,RW,DZ(1),...,DZ(n),AZ(n),RZ(n),UZ(n)) gilt dann

(3)

$$z \in \bar{N}_{WMAX} \times N_n \times \prod_{i=1}^{n} (N_{DZM(i)} \times N_{AZM(i)} \times N_{RZM(i)} \times N_{UZM(i)}) =: Z.$$

Z ist also eine endliche Menge.

Die Menge E der möglichen Signale braucht nicht verändert zu werden; es können noch dieselben Signale eintreffen. Die Zustandsmenge des zu unserem Modell gehörigen Entscheidungsmodells, Z × E, ist ebenfalls endlich.

Die Wahrscheinlichkeitsgesetze, nach denen die Signale eintreffen, können wir im Prinzip übernehmen. Da wir sie aber nur von den Zuständen abhängen lassen wollen, können zum Beispiel die Ankunftswahrscheinlichkeiten $P_i(k)$ für $k > DZM(i)$ nicht mehr berücksichtigt werden, sie werden vielmehr durch $P_i(DZM(i))$ ersetzt. Etwas nachlässig ausgedrückt heißt das, daß man die "Schwänze" der Verteilungen durch geometrische Verteilungen ersetzt hat. Für die Endewahrscheinlichkeiten gilt das Analoge bezüglich der Rechenzeitverteilungen.

Für die Kosten gilt ebenfalls, daß sie nur von den Zuständen abhängen sollen. Der für sie jeweils maßgebende Parameter bleibt also oberhalb der Schranken konstant. Der Zuwachs der Gesamtkosten über eine längere Zeit erfolgt daher schließlich linear mit der Zeit.

Die Beschränkung des Zustandsraumes erfordert nun eine Deformation der Wahrscheinlichkeitsverteilungen und der Kostenfunktionen.

Die in (5.1) angegebene Wirkung der Aktionen würde bei unserem beschränkten Zustandsraum möglicherweise auf nicht mehr zulässige Zustände führen. Um dieses zu vermeiden, müssen wir die Gleichungen (16) bis (19) aus (5.1) abändern. Die neuen Vorschriften lauten nun:

$$(4)\ DZ(i)' = \begin{cases} 0, \text{ falls } (DZ(i)=0 \wedge RW \neq i) \vee (DZ(i)=0 \wedge RW=i \wedge ES=0) \vee \\ \qquad\qquad \vee (DZ(i)>0 \wedge AS(i)=1), \\ 1, \text{ falls } DZ(i)=0 \wedge ES=1 \wedge RW=i \\ \min(DZ(i)+1, DZM(i)), \text{ falls } DZ(i)>0 \wedge AS(i)=0 \end{cases}$$

$$(5)\ AZ(i)' = \begin{cases} 0, \text{ falls } (AZ(i)=0 \wedge AS(i)=0) \vee (AZ(i)>0 \wedge RW=i \wedge ES=1) \\ 1, \text{ falls } AZ(i)=0 \wedge AS(i)=1. \\ \min(AZ(i)+1, AZM(i)) \text{ für } (AZ(i)>0 \wedge RW \neq i) \vee (AZ(i)> \\ \qquad\qquad >0 \wedge RW=i \wedge ES=0). \end{cases}$$

$$(6)\ RZ(i)' = \begin{cases} 0, & \text{falls } AZ(i)'=0, \\ RZ(i), & \text{falls } AZ(i)'>0 \wedge k \neq i \\ \min(RZ(i)+1, RZM(i)), & \text{falls } AZ(i)'>0 \wedge k=i, \end{cases}$$

$$(7)\ UZ(i)' = \begin{cases} 0, & \text{falls } AZ(i)'=0 \vee RW'=i \\ \min(UZ(i)+1, UZM(i)), & \text{falls } AZ(i)'>0 \wedge RW' \neq i. \end{cases}$$

Mit diesen Änderungen haben wir ein endliches Modell erhalten. Wir können nun optimale Politiken in endlich vielen Schritten bestimmen.

Betrachten wir etwa einen endlichen Planungshorizont, so führt uns die Optimalitätsgleichung (7) aus (4.2) zum Ziel. Bemerkt sei hier noch, daß ein Entscheidungsmodell mit endlichem Planungshorizont natürlich nicht mehr stationär ist. Die aus der Optimalitätsgleichung konstruierte optimale Po-

litik ist also im allgemeinen auch nicht stationär.
Bei einem unendlichen Planungshorizont mit einem Diskontfaktor $\beta < 1$ ist das Verfahren von Howard (1960), das wir für diesen Fall in (4.3) beschrieben haben, anwendbar.

Für den Fall $\beta = 1$ stehen ebenfalls Iterationsverfahren von Howard zur Verfügung, die nach endlich vielen Schritten eine Politik liefern, die bezüglich der mittleren Kosten pro Übergang optimal ist (vgl. 4.3)).

Bei praktisch interessanten Anwendungen wird die Anzahl der zu berücksichtigenden Zustände schon sehr groß, so daß man bei der Programmierung der Optimierungsverfahren bei den verfügbaren Rechenanlagen auf Speicherplatzschwierigkeiten stößt. Wir beschränken uns daher in (5.4) auf ein Modell mit zwei Bedienungsplätzen, das aber zur Demonstration einiger interessanter Punkte ausreicht.

5.4 Zwei-Benutzer-Modell

Es soll nun das im vorigen Abschnitt beschriebene Modell für zwei Benutzer behandelt werden. Es seien zusätzlich DZM(i)=AZM(i)=RZM(i)=UZM(i) = 1 für i=1,2 . Wir betrachten also nur geometrische Verteilungen für die Denkzeiten und die Rechenzeiten, die wir jetzt durch die Parameter P_1, P_2 bzw. Q_1, Q_2 beschreiben können. Bei den Kosten soll zwischen Unterbrechungszeitkosten und Verzögerungskosten nicht mehr unterschieden werden. Unter diesen Umständen genügen zur Beschreibung des Zustands des Systems die vier Parameter W($1 \leq W \leq$ WMAX) für die Weckerstellung, RW(=0,1,2) für die Belegung des Rechnerkerns und K1,K2 mit Ki=1, falls ein Auftrag von B_i vorliegt, Ki=0 sonst für i=1,2.

An Kosten haben wir noch KN, die Kosten für die Bearbeitung des Weckersignals, KA1 und KA2, die Anwesenheitskosten für Aufträge von B_1 bzw. B_2, KW1 und KW2, die Kosten für die Verzögerung eines Auftrags von B_1 bzw. B_2 um ein Zeitintervall und die Kosten KU1 und KU2 für die Unterbrechung eines Auftrags. Auf der folgenden Seite werden die Bezeichnungen noch einmal übersichtlich zusammengestellt.

Seien die Zustände z_t=(W,RW,K1,K2) und z_{t+1}=(WN,RWN,K1N,K2N) mit i bzw. j bezeichnet, so sind die Gesamtkosten im Intervall (t,t+1)

(1) $$r_{i,j} = \delta_{1,W}\cdot KN + \delta_{1,K1N}\cdot KA1 + \delta_{1,K2N}\cdot KA2 + \delta_{1,K1N}\cdot \cdot(1-\delta_{1,RWN})KW1 + \delta_{1,K2N}\cdot(1-\delta_{2,RWN})\cdot KW2 + \delta_{1,RW}\cdot \cdot\delta_{1,K1N}\cdot(1-\delta_{1,RWN})KU1+\delta_{2,RW}\cdot\delta_{1,K2N}\cdot(1-\delta_{2,RWN})\cdot KU2.$$

Zusammenstellung der wichtigsten Bezeichnungen

Zustandsparameter:

W (WN) für die Weckerstellung,
RW (RWN) für die Rechnerkernbelegung,
K1 (K1N) für die Anwesenheit eines Auftrags von B_1
K2 (K2N) für die Anwesenheit eines Auftrags von B_2
WMAX Maximalwert von W

Wahrscheinlichkeiten:

P_1 für die Ankunft eines Auftrags von B_1, falls von dort kein unbearbeiteter Auftrag mehr ansteht;
P_2 für die Ankunft eines Auftrags von B_2, falls von dort kein unbearbeiteter Auftrag mehr ansteht;
Q_1 für die Beendigung eines Auftrags von B_1, falls er bearbeitet wird;
Q_2 für die Beendigung eines Auftrags von B_2, falls er bearbeitet wird

Kosten:

KN für die Verarbeitung eines Weckersignals,
KU1 für die Unterbrechung eines Auftrags von B_1,
KU2 für die Unterbrechung eines Auftrags von B_2,
KA1 für eine Zeiteinheit der Anwesenheit eines Auftrags von B_1,
KA2 für eine Zeiteinheit der Anwesenheit eines Auftrags von B_2,
KW1 für eine Zeiteinheit der Verzögerung eines Auftrags von B_1,
KW2 für eine Zeiteinheit der Verzögerung eines Auftrags von B_2.

Die Betriebssignale setzen sich wieder aus dem Weckersignal, dem Endesignal und den Ankunftssignalen von den Bedienungsplätzen zusammen.

Dieses Modell soll bei einem unendlichen Planungshorizont ohne Diskontierung mit Hilfe des Howardschen Algorithmus aus (4.3) behandelt werden. Um Pathologien in der Struktur der zu einer Politik gehörenden Markovschen Kette zu vermeiden, setzen wir voraus, daß $0 < P_i < 1$ und $0 < Q_i < 1$ für $i=1,2$ und daß der Rechnerkern nicht untätig bleiben darf (RW=0), wenn mindestens ein Auftrag zur Bearbeitung ansteht.

Dann gilt für eine beliebige Politik:

a) Die Ein-Schritt-Übergangswahrscheinlichkeit vom Zustand (1,0,0,0) in denselben Zustand ist positiv. Steht nämlich kein Auftrag an und kommt im Verlaufe eines Intervalls kein neuer Auftrag an, so muß der neue Zustand jedenfalls wieder (1,0,0,0) sein. Die Wahrscheinlichkeit für dieses Ereignis beträgt $(1-P_1)(1-P_2) > 0$.

b) Der Zustand (1,0,0,0) wird mit positiver Wahrscheinlichkeit von jedem anderen Zustand aus erreicht.

Für (1,0,0,0) ist das die Behauptung a). Sei nun (W,1,1,0). Ein Auftrag von B_1 wird also vom Rechnerkern bearbeitet. Mit der Wahrscheinlichkeit $Q_1(1-P_2)$ wird dieser Auftrag im Verlaufe des Intervalls beendet, ohne daß ein neuer Auftrag von B_2 eintrifft. Diese Wahrscheinlichkeit ist aber positiv nach Voraussetzung. Der neue Zustand muß in diesem Fall natürlich (1,0,0,0) sein. Ganz analog schließt man für die Ein-Schritt-Übergangswahrscheinlichkeit von (W,2,0,1) nach (1,0,0,0).

Dagegen sind die Ein-Schritt-Übergangswahrscheinlichkeiten von (W,k,1,1) für k=1,2 nach (1,0,0,0) gleich 0, da ja in einem Intervall höchstens der bearbeitete Auftrag beendet werden kann, nicht aber der andere. Dafür ist aber die Zwei-Schritt-Übergangswahrscheinlichkeit positiv, nämlich gleich $Q_1 \cdot Q_2 \cdot (1-P_1)$ bzw. $Q_1 \cdot Q_2 \cdot (1-P_2)$. Es müssen doch beide Aufträge in den beiden aufeinander folgenden Intervallen beendet werden, wobei im zweiten Intervall der im ersten beendete Auftrag nicht bereits durch einen neuen Auftrag ersetzt werden darf.

Aus den Aussagen a) und b) folgert man aber sofort (vgl.z.B. Doob 1953, V,3), daß zu jeder Politik eine Markovsche Kette mit nur einer ergodischen Klasse gehört, die außerdem noch aperiodisch ist. Damit ist die Voraussetzung für die Anwendbarkeit des Howardschen Verfahrens erfüllt.

Für die Durchführung des Verfahrens ist es notwendig, die durch (W,RW,K1,K2) definierten Zustände linear anzuordnen. Das geschieht durch die folgende Numerierungsvorschrift:

$$(2) \qquad NR(W,RW,K1,K2) = \begin{cases} 2 \cdot K1 + K2 + 1 \text{ , falls } RW = 0 \\ 4 \cdot W + K1 + K2 + 2 \cdot (RW-1) \text{ sonst.} \end{cases}$$

Dabei wird vorausgesetzt, daß (W,RW,K1,K2) einen zulässigen Zustand beschreibt (die Zustände 2,3,4 gelten auch noch als zulässig, obwohl sie von keiner Politik gewählt werden dürfen!).

Die Gesamtzahl der auftretenden Zustände bei vorgegebener maximalen Zeitvorgabe WMAX beträgt

$$(3) \qquad N = 4 \cdot (WMAX + 1).$$

Aus der Nummer i eines Zustands kann man die Werte der Parameter W, RW, K1, K2 zurückgewinnen. Es gilt:
Für $i \leq 4$:

$$
\begin{aligned}
W &= 1 \quad \text{für } i \leq 4, \\
RW &= 0 \quad \text{für } i \leq 4, \\
K1 &= \begin{cases} 0 & \text{für } i = 1,2 \\ 1 & \text{für } i = 3,4 \end{cases} \\
K2 &= \begin{cases} 0 & \text{für } i = 1,3 \\ 1 & \text{für } i = 2,4 \end{cases}
\end{aligned} \tag{4}
$$

Für i größer als 4 erhält man:

$$
\begin{aligned}
W &= \text{entier } ((i-1)/4) \\
RW &= \begin{cases} 1, & \text{falls } i - 4 \cdot W < 3, \\ 2, & \text{falls } i - 4 \cdot W \geq 3, \end{cases} \\
K1 &= \begin{cases} 1, & \text{falls RW = 1 oder } i - 4 \cdot W = 4, \\ 0 & \text{sonst;} \end{cases} \\
K2 &= \begin{cases} 1, & \text{falls RW=2 oder } i - 4 \cdot W = 2, \\ 0 & \text{sonst.} \end{cases}
\end{aligned} \tag{5}
$$

Für die in dieser Weise numerierten Zustände findet man die folgenden Übergangskosten:

$$r_{i,j} = \sum_{l=1}^{7} k_l(i,j) \text{ mit } i,j=1,2,\ldots,4 \cdot WMAX + 4; \tag{6}$$

darin sind die Weckerkosten

$$k_1(i,j) = \begin{cases} KN & \text{für } i=1,2,3,4,5,6,7,8 \\ 0 & \text{sonst ,} \end{cases} \tag{7}$$

die Unterbrechungskosten

$$k_2(i,j) = \begin{cases} KU1 & \text{falls } i=5 \text{ oder } 6 \text{ und } j=3,4,4 \cdot k+4 \\ & (1 \leq k \leq WMAX), \\ 0 & \text{sonst,} \end{cases} \tag{8}$$

$$(9)\quad k_3(i,j) = \begin{cases} KU2 & \text{falls } i=7,8 \text{ und } j=2,4,4\cdot k+2 \ (1\leq k\leq WMAX), \\ 0 & \text{sonst;} \end{cases}$$

die Wartekosten

$$(10)\quad k_4(i,j) = \begin{cases} KW1, & \text{falls } j=3,4,4\cdot k+4 \ (1\leq k\leq WMAX), \\ 0 & \text{sonst.} \end{cases}$$

$$(11)\quad k_5(i,j) = \begin{cases} KW2, & \text{falls } j=2,4,4\cdot k+2 \ (1\leq k\leq WMAX), \\ 0 & \text{sonst;} \end{cases}$$

und den Anwesenheitskosten

$$(12)\quad k_6(i,j) = \begin{cases} KA1 & \text{für } j=3,4,4\cdot k+1,4k+2,4k+4 \ (1\leq k\leq WMAX), \\ 0 & \text{sonst,} \end{cases}$$

$$(13)\quad k_7(i,j) = \begin{cases} KA2 & \text{für } j=2,4,4k+2,4k+3,4k+4 \ (1\leq k\leq WMAX), \\ 0 & \text{sonst.} \end{cases}$$

In einem Zustand i können maximal vier der möglichen Signale mit positiver Wahrscheinlichkeit eintreffen. Bei jedem dieser Signale muß in Abhängigkeit vom Zustand eine Entscheidung getroffen werden, wobei jedoch in einigen Fällen diese Entscheidung der Sache nach bereits eindeutig festgelegt ist. Die Signale in einem Zustand i, zu denen die Entscheidung nicht schon von vornherein festliegt, seien numeriert. Ihre Anzahl sei n_i. Dem j-ten Signal dieser Art im Zustand i werde eine Entscheidungsvariable d(i,j) zugeordnet, die ganzzahlige Werte 1,...,WMAX und eventuell -1,-2,....,-WMAX annehmen kann. Die folgende Tabelle beschreibt die Folgezustände, die bei den einzelnen Werten der Entscheidungsvariablen eintreffen. Außerdem sind die Wahrscheinlichkeiten der jeweils mit positiver Wahrscheinlichkeit auftretenden Signale angegeben. In der mit SGNL überschriebenen Spalte bedeutet eine 1 in der ersten

Spalte das Auftreten des Weckersignals, in der zweiten Spalte das Eintreffen eines Endesignals vom Rechnerkern, in der dritten das Eintreffen eines neuen Auftrags an der Bedienungsstation B_1 und in der letzten Spalte das Eintreffen eines neuen Auftrags von B_2 . Eine 0 stellt jeweils das Ausbleiben des entsprechenden Signals dar. Weiter ist M := WMAX.

Zustand	SGNL	Wahrscheinlichkeit	Entscheidungsvariable	Folgezustand für d(i,j)= -1	1	...	-k	k	...	-M	M
	1000	$(1-P_1)(1-P_2)$	-	1							
1	1010	$P_1(1-P_2)$	d(1,1)	-	5	...	-	4k+1	...	-	4M+1
	1001	$(1-P_1)P_2$	d(1,2)	-	7	...	-	4k+3	...	-	4M+3
	1011	P_1P_2	d(1,3)	6	8	...	4k+2	4k+4	...	4M+2	4M+4
2	1000	$(1-P_1)$	d(2,1)	-	7	...	-	4k+3	...	-	4M+3
	1010	P_1	d(2,2)	6	8	...	4k+2	4k+4	...	4M+2	4M+4
3	1000	$(1-P_2)$	d(3,1)	-	5	...	-	4k+1	...	-	4M+1
	1001	P_2	d(3,2)	6	8	...	4k+2	4k+4	...	4M+2	4M+4
4	1000	1	d(4,1)	6	8	...	4k+2	4k+4	...	4M+2	4M+4
	1100	$(1-P_2)Q_1$	-	1							
5	1000	$(1-P_2)(1-Q_1)$	d(5,1)	-	5	...	-	4k+1	...	-	4M+1
	1001	$P_2(1-Q_1)$	d(5,2)	6	8	...	4k+2	4k+4	...	4M+2	4M+4
	1101	P_2Q_1	d(5,3)	-	7	...	-	4k+3	...	-	4M+3
6	1000	$(1-Q_1)$	d(6,1)	6	8	...	4k+2	4k+4	...	4M+2	4M+4
	1100	Q_1	d(6,2)	-	7	...	-	4k+3	...	-	4M+3

Zustand	SGNL	Wahrscheinlichkeit	Entscheidungsvariable	Folgezustand für d(i,j)= -1 1 ... -k k ... -M M
7	1000	$(1-P_1)(1-Q_2)$	d(7,1)	- 7 ... - 4k+3 ... - 4M+3
	1100	$(1-P_1)Q_2$	-	1
	1010	$P_1(1-Q_2)$	d(7,2)	6 8 ... 4k+2 4k+4 ... 4M+2 4M+4
	1110	P_1Q_2	d(7,3)	- 5 ... - 4k+1 ... - 4M+1
8	1000	$(1-Q_2)$	d(8,1)	6 8 ... 4k+2 4k+4 ... 4M+2 4M+4
	1100	Q_2	d(8,2)	- 5 ... - 4k+1 ... - 4M+1
	0100	$Q_1(1-P_2)$	-	1
4k'+1	0000	$(1-Q_1)(1-P_2)$	-	4(k'-1)+1
(1<k')	0001	$(1-Q_1)P_2$	-	4(k'-1)+2
	0101	Q_1P_2	d(4k'+1,1)	- 7 ... - 4k+3 ... - 4M+3
4k'+2	0000	$1-Q_1$	-	4(k'-1)+2
(1<k')	0100	Q_1	d(4k'+2,1)	- 7 ... - 4k+3 ... - 4M+3
	0100	$Q_2(1-P_1)$	-	1
4k'+3	0000	$(1-Q_2)(1-P_1)$	-	4(k'-1)+3
(1<k')	0010	$P_1(1-Q_2)$	-	4(k'-1)+4
	0110	P_1Q_2	d(4k'+3,1)	- 5 ... - 4k+1 ... - 4M+1
4k'+4	0000	$1-Q_2$	-	4(k'-1)+4
(1<k')	0100	Q_2	d(4k'+4,1)	- 5 ... - 4k+1 ... - 4M+1

Sei nun eine Politik festgelegt, indem jede Entscheidungsvariable d(i,j)(i=1,2,...,N; j=1,2,...,n_i) einen zulässigen Wert erhalten hat. Zur Vereinfachung setzen wir noch

(14) $$d_k(i,j) = \begin{cases} 1, & \text{falls } d(i,j) = k \\ 0 & \text{sonst.} \end{cases}$$

Darin sind $i=1,2,\ldots,N$; $j=1,2,\ldots,n_i$; $k=+1,-1,+2,-2,\ldots,+M,-M$. Dann liest man aus der vorhergehenden Tabelle die folgenden Übergangswahrscheinlichkeiten ab für $l = 1,2,\ldots,M$:

$$p_{1,1} = (1-P_1)(1-P_2),$$

$$p_{1,2} = p_{1,3} = p_{1,4} = 0,$$

$$p_{1,4l+1} = P_1(1-P_2)d_1(1,1),$$

$$p_{1,4l+2} = P_1P_2d_{-1}(1,3),$$

$$p_{1,4l+3} = P_2(1-P_1)d_1(1,2),$$

$$p_{1,4l+4} = P_1P_2d_1(1,3);$$

$i=2$: $p_{2,1} = p_{2,2} = p_{2,3} = p_{2,4} = 0,$

$p_{2,4l+1} = 0,\ p_{2,4l+2} = P_1d_{-1}(2,2),$

$p_{2,4l+3} = (1-P_1)d_1(2,1),$

$p_{2,4l+4} = P_1\, d_1(2,2);$

$i=3$: $p_{3,1} = p_{3,2} = p_{3,3} = p_{3,4} = p_{3,4l+3} = 0,$

$p_{3,4l+1} = (1-P_2)d_1(3,1),\ p_{3,4l+2} = P_2d_{-1}(3,2),$

$p_{3,4l+4} = P_2\, d_1(3,2);$

$i=4$: $p_{4,1} = p_{4,2} = p_{4,3} = p_{4,4} = p_{4,4l+1} = p_{4,4l+3} = 0,$

$p_{4,4l+2} = d_{-1}(4,1),\ p_{4,4l+4} = d_1(4,1);$

$i=5$: $p_{5,1} = (1-P_2)Q_1,\ p_{5,2} = p_{5,3} = p_{5,4} = 0,$

$p_{5,4l+1} = (1-P_2)(1-Q_1)d_1(5,1),$

$p_{5,4l+2} = P_2(1-Q_1)d_{-1}(5,2),\ p_{5,4l+3} = P_2Q_1\, d_1(5,3),$

$p_{5,4l+4} = P_2(1-Q_1)d_1(5,2);$

$i=6$: $p_{6,1} = p_{6,2} = p_{6,3} = p_{6,4} = p_{6,4l+1} = 0$,

$p_{6,4l+2} = (1-Q_1)d_{-1}(6,1)$, $p_{6,4l+3} = Q_1 d_1(6,2)$,

$p_{6,4l+4} = (1-Q_1)\, d_1(6,1)$;

$i=7$: $p_{7,1} = (1-P_1)Q_2$, $p_{7,2} = p_{7,3} = p_{7,4} = 0$,

$p_{7,4l+1} = P_1 Q_2 d_1(7,3)$, $p_{7,4l+2} = P_1(1-Q_2)d_{-1}(7,2)$,

$p_{7,4l+3} = (1-P_1)(1-Q_2)d_1(7,1)$, $p_{7,4l+4} = P_1(1-Q_2)d_1(7,2)$;

$i=8$: $p_{8,1} = p_{8,2} = p_{8,3} = p_{8,4} = p_{8,4l+3} = 0$,

$p_{8,4l+1} = Q_2 d_1(8,2)$, $p_{8,4l+2} = (1-Q_2)d_{-1}(8,1)$,

$p_{8,4l+4} = (1-Q_2)d_1(8,1)$;

$i=4k+1$ $(1 < k)$:

$p_{4k+1,1} = Q_1(1-P_2)$, $p_{4k+1,2} = p_{4k+1,3} = p_{4k+1,4} = 0$,

$p_{4k+1,4l+1} = (1-Q_1)(1-P_2)\, \delta_{1,k-1}$,

$p_{4k+1,4l+2} = (1-Q_1)P_2\, \delta_{1,k-1}$,

$p_{4k+1,4l+3} = P_2 Q_1\, d_1(4k+1,1)$,

$p_{4k+1,4l+4} = 0$;

$i=4k+2$ $(1 < k)$:

$p_{4k+2,1} = p_{4k+2,2} = p_{4k+2,3} = p_{4k+2,4} = p_{4k+2,4l+1} = 0$,

$p_{4k+2,4l+2} = (1-Q_1)\, \delta_{1,k-1}$,

$p_{4k+2,4l+3} = Q_1 d_1(4k+2,1)$, $p_{4k+2,4l+4} = 0$;

$i=4k+3$ $(1 < k)$:

$p_{4k+3,1} = Q_2(1-P_1)$, $p_{4k+3,2} = p_{4k+3,3} = p_{4k+3,4} = 0$,

$p_{4k+3,4l+1} = P_1 Q_2 d_1(4k+3,1)$, $p_{4k+3,4l+2} = 0$,

$$p_{4k+3,4l+3} = (1-Q_2)(1-P_1)\ \delta_{l,k-1},$$

$$p_{4k+4,4l+4} = (1-Q_2)P_1\ \delta_{l,k-1};$$

$i=4k+4$ $(1 < k)$:

$$p_{4k+4,1} = p_{4k+4,2} = p_{4k+4,3} = p_{4k+4,4} = 0,$$

$$p_{4k+4,4l+1} = Q_2 d_1(4k+4,1),\ p_{4k+4,4l+2} = 0,$$

$$p_{4k+4,4l+3} = 0,\ p_{4k+4,4l+4} = (1-Q_2)\ \delta_{l,k-1}\ .$$

Auf diese Weise haben wir die Übergangswahrscheinlichkeiten der Markovschen Kette, die zu einer Politik gehört, explizit angegeben.

Aus den in der obigen Tabelle angegebenen möglichen Werten der Entscheidungsvariablen entnimmt man sofort, daß die Anzahl der möglichen Politiken gleich

(15) $$2^8 \cdot M^{4M+14}$$

ist.

Die erwarteten unmittelbaren Erlöse q_i $(i=1,\ldots,4M+4)$ sind oben definiert worden als

(16) $$-q_i = \sum_{j=1}^{4M+4} p_{ij}\ r_{ij}\ .$$

Die q_i können wir jetzt unmittelbar hinschreiben (die auftretenden Summen gehen über l von 1 bis M):

$$q_1 = KN + P_1P_2\ KW1 \sum d_1(1,3) + P_1P_2\ KW2 \sum d_{-1}(1,3) +$$

$$+ KA1\ P_1 + KA2\ P_2\ ;$$

$$q_2 = KN + P_1\,KW1 \sum d_1(2,2) + P_1\,KW2 \sum d_{-1}(2,2) + P_1\,KA1 + KA2;$$

$$q_3 = KN + P_2\,KW1 \sum d_1(3,2) + P_2\,KW2 \sum d_{-1}(3,2) + KA1 + P_2\,KA2;$$

$$q_4 = KN + KW1 \sum d_1(4,1) + KW2 \sum d_{-1}(4,1) + KA1 + KA2;$$

$$q_5 = KN + KU1\,P_2\,(1-Q_1) \sum d_1(5,2) + KW1\,P_2\,(1-Q_1) \sum d_1(5,2) + KW2\,P_2\,(1-Q_1) \sum d_{-1}(5,2) + KA1\,(1-Q_1) + KA_2\,P_2;$$

$$q_6 = KN + KU1\,(1-Q_1) \sum d_1(6,1) + KW1\,(1-Q_1) \sum d_1(6,1) + KW2\,(1-Q_1) \sum d_{-1}(6,1) + KA1\,(1-Q_1) + KA2;$$

$$q_7 = KN + KU2\,(1-Q_2)\,P_1 \sum d_{-1}(7,2) + KW1\,(1-Q_2)\,P_1 \sum d_1(7,2) + KW2\,(1-Q_2)\,P_1 \sum d_{-1}(7,2) + KA1\,P_1 + KA2\,(1-Q_2);$$

$$q_8 = KN + KU2\,(1-Q_2) \sum d_{-1}(8,1) + KW1\,(1-Q_2) \sum d_1(8,1) + KW2\,(1-Q_2) \sum d_{-1}(8,1) + KA1 + KA2\,(1-Q_2);$$

Für k=2,3,...M erhält man:

$$q_{4k+1} = KW2\,(1-Q_1)\,P_2 + KA1\,(1-Q_1) + KA2\,P_2;$$

$$q_{4k+2} = KW2\,(1-Q_1) + KA1\,(1-Q_1) + KA2;$$

$$q_{4k+3} = KW1\,(1-Q_2)P_1 + KA_1\,P_1 + KA2\,(1-Q_2);$$

$$q_{4k+4} = KW1\,(1-Q_2) + KA1 + KA2\,(1-Q_2).$$

Seien nun $\tilde{\pi}_1, \tilde{\pi}_2, \ldots, \tilde{\pi}_N$ die zur betrachteten Politik gehörigen stationären Grenzwahrscheinlichkeiten. Dann hat der Gewinn pro Schritt die Gestalt:

(17)

$$
\begin{aligned}
-G = {} & \sum_{i=1}^{8} \tilde{\pi}_i \; KN + [\pi_5 \, P_2 \, (1-Q_1) \sum_{l=1}^{M} d_1(5,2) + \\
& + \tilde{\pi}_6 \, (1-Q_1) \sum_{l=1}^{M} d_1(6,1)] \; KU1 + [\tilde{\pi}_7 \, P_1 \, (1-Q_2) \sum_{l=1}^{M} d_{-1}(7,2) \\
& + \tilde{\pi}_8 \, (1-Q_2) \sum_{l=1}^{M} d_{-1}(8,1)] \; KU2 + [\tilde{\pi}_1 \, P_1 \, P_2 \sum_{l=1}^{M} d_1(1,3) + \\
& + \tilde{\pi}_5 \, P_2 \, (1-Q_1) \sum_{l=1}^{M} d_1(5,2) + \tilde{\pi}_6 \, (1-Q_1) \sum_{l=1}^{M} d_1(6,1) + \\
& + \tilde{\pi}_7 \, P_1 \, (1-Q_2) \sum_{l=1}^{M} d_1(7,2) + \tilde{\pi}_8 \, (1-Q_2) \sum_{l=1}^{M} d_1(8,1) + \\
& + \sum_{k=2}^{M} \{\tilde{\pi}_{4k+3} \, (1-Q_2) \, P_1 + \tilde{\pi}_{4k+4} \, (1-Q_2)\}] \; KW1 + \\
& + [\tilde{\pi}_1 \, P_1 \, P_2 \sum_{l=1}^{M} d_{-1}(1,3) + \tilde{\pi}_5 \, P_2 \, (1-Q_1) \sum_{l=1}^{M} d_{-1}(5,2) + \\
& + \tilde{\pi}_6 \, (1-Q_1) \sum_{l=1}^{M} d_{-1}(6,1) + \tilde{\pi}_7 \, (1-Q_2) \, P_1 \sum_{l=1}^{M} d_{-1}(7,2) + \\
& + \tilde{\pi}_8 \, (1-Q_2) \sum_{l=1}^{M} d_{-1}(8,1) + \sum_{k=2}^{M} \{\tilde{\pi}_{4k+1}(1-Q_1) \, P_2 + \\
& + \tilde{\pi}_{4k+2} \, (1-Q_1)\}] \; KW2 + [\tilde{\pi}_1 \, P_1 + \sum_{k=1}^{M} \{(1-Q_1)(\tilde{\pi}_{4k+1} + \\
& + \tilde{\pi}_{4k+2}) + \tilde{\pi}_{4k+3} \cdot P_1 + \tilde{\pi}_{4k+4}\}] \; KA1 + [\tilde{\pi}_1 \, P_2 + \\
& + \sum_{k=1}^{M} \{P_2 \, \tilde{\pi}_{4k+1} + \tilde{\pi}_{4k+2} + (\tilde{\pi}_{4k+3} + \tilde{\pi}_{4k+4})(1-Q_2)\}] \; KA2.
\end{aligned}
$$

Die stationären Grenzwahrscheinlichkeiten $\tilde{\pi}_1, \tilde{\pi}_2, \ldots, \tilde{\pi}_N$ hängen selbstverständlich von der Politik ab.

Aus der soeben hergeleiteten Gestalt des Gewinns ziehen wir einige einfache Folgerungen: Es sei $\varkappa$ = (KN,KA1,KA2,KW1,KW2, KU1,KU2) ein Vektor im R^7. Die Koeffizienten der Kosten im Gewinn sind Konstante, die nur von der Politik abhängen. Dann gilt:

(18) Ist $\mathfrak{p}^*$ eine optimale Politik für $\varkappa$, dann ist $\mathfrak{p}^*$ auch eine optimale Politik für $\alpha\varkappa$, für $\alpha \geq 0$.

(19) Ist $\mathfrak{p}^*$ eine optimale Politik für $\varkappa_1$ und $\varkappa_2$, so ist $\mathfrak{p}^*$ auch eine optimale Politik für $\alpha\varkappa_1 + (1-\alpha)\varkappa_2$ für $0 \leq \alpha \leq 1$.

Der Beweis dieser Aussagen ist trivial.

Die Menge der Kostenvektoren, für die eine gegebene Politik optimal ist, bildet also einen konvexen Kegel mit der Spitze in (0,0,0,0,0,0,0). Aus der Stetigkeit des Gewinns in den Kostenparametern folgert man ebenfalls leicht, daß dieser Kegel abgeschlossen ist.

Die Abhängigkeit des Gewinns von den Wahrscheinlichkeitsparametern ist nicht so einfach. Aus der Wertbestimmung des Howardschen Optimierungsverfahrens für den nichtdiskontierten Fall jedoch erhalten wir das Gleichungssystem:

$$
\begin{aligned}
&(1-p_{11})\, v_1 - p_{12}\, v_2 - \dots - p_{1,N-1}\, v_{N-1} + G = q_1\\
&-p_{21}\, v_1 + (1-p_{22})\, v_2 - \dots - p_{2,N-1}\, v_{N-1} + G = q_2\\
&\dots\dots\dots

Nach unseren Voraussetzungen ist dieses Gleichungssystem eindeutig lösbar. Nach der Cramerschen Regel erhalten wir:

$$(21)\quad G = \frac{\begin{vmatrix} 1-p_{11} & -p_{12} & \dots & \dots & -p_{1,N-1} & q_1 \\ -p_{21} & 1-p_{22} & \dots & \dots & -p_{2,N-1} & q_2 \\ \dots & \dots & \dots & \dots & \dots & \dots \\ -p_{N-1,1} & -p_{N-1,2} & \dots & \dots & 1-p_{N-1,N-1} & q_{N-1} \\ -p_{N,1} & -p_{N,2} & \dots & \dots & -p_{N,N-1} & q_N \end{vmatrix}}{\begin{vmatrix} 1-p_{11} & -p_{12} & \dots & \dots & -p_{1,N-1} & 1 \\ -p_{21} & 1-p_{22} & \dots & \dots & -p_{2,N-1} & 1 \\ \dots & \dots & \dots & \dots & \dots & \dots \\ -p_{N-1,1} & -p_{N-1,2} & \dots & \dots & 1-p_{N-1,N-1} & 1 \\ -p_{N,1} & -p_{N,2} & \dots & \dots & -p_{N,N-1} & 1 \end{vmatrix}}$$

Die Übergangswahrscheinlichkeiten und die erwarteten unmittelbaren Erlöse sind lineare Funktionen in jedem der Wahrscheinlichkeitsparameter P_1, P_2, Q_1, Q_2. Aus (21) erkennen wir also, daß der Gewinn eine rationale Funktion eines jeden Wahrscheinlichkeitsparameters ist.

Das hätten wir auf ähnliche Weise aus der Gestalt (17) für den Gewinn schließen können, wenn wir die stationären Grenzwahrscheinlichkeiten $\tilde{\pi}_1, \tilde{\pi}_2, \dots, \tilde{\pi}_N$ aus dem Gleichungssystem

$$(22)\quad \begin{aligned} &\sum_{i=1}^{N} \tilde{\pi}_i\, p_{ij} = \tilde{\pi}_j \quad (j=1,2,\dots,N) \\ &\tilde{\pi}_1 + \tilde{\pi}_2 + \dots + \tilde{\pi}_N = 1 \end{aligned}$$

nach der Cramerschen Regel dargestellt und in (17) eingesetzt hätten.

Jetzt wollen wir uns noch etwas eingehender mit den Politiken beschäftigen. Unter den formal möglichen Politiken gibt es einige, die intuitiv besonders leicht zu verstehen sind. Zunächst sind da die Politiken zu nennen, die die Aufträge streng in der Reihenfolge ihres Eintreffens abarbeiten (first in first out). In unserem Modell sind im Zustand 1 die Aufträge, die im gleichen Intervall eintreffen, hinsichtlich ihrer zeitlichen Reihenfolge nicht unterscheidbar. In diesem Fall kann der Zuteilungsalgorithmus entweder den Auftrag von B_1 oder denjenigen von B_2 bevorzugen. Wir erhalten daher die beiden Politiken FIFO(1) und FIFO(2). Im Fall einer Rechnerkernvergabe werde von FIFO immer die maximal zulässige Zeitquantenzahl zugewiesen.

Eine andere Klasse von Politiken erhält man, wenn man fordert, daß immer der jeweils zuletzt eingetroffene Auftrag zu Ende bearbeitet wird (last in first out). Auch hier gibt es Situationen, in denen die Reihenfolge nicht erkennbar ist. Entsprechend gibt es die Politiken LIFO(1) und LIFO(2). Der erste eintreffende Auftrag erhält dabei ein Zeitquantum, damit beim Eintreffen des zweiten Auftrags sofort auf den zuletzt gekommenen umgeschaltet werden kann, der dann natürlich die maximal zulässige Quantenzahl erhält. Man kann nun diese strengen LIFO-Politiken etwas abmildern, indem man dem zuerst eingetroffenen Auftrag mehr als nur ein Quantum Zeitvorgabe gewährt. Solche Politiken wollen wir mit LIFO(k,1,1) und LIFO(k,1,2) bezeichnen, wenn der zuerst eintreffende Auftrag k(1) Zeitquanten erhält, falls er von B_1 (B_2) kommt.

Natürlich gilt LIFO(1) = LIFO(1,1,1) und LIFO(1,1,2)=LIFO(2), so daß die früheren Bezeichnungen entbehrlich sind.
Die FIFO- und LIFO-Politiken bevorzugen Aufträge aufgrund ihres Ankunftszeitpunktes. Eine Bevorzugung eines Auftrags kann jedoch auch in Abhängigkeit von der Bedienungsstation, von der er kommt, erfolgen. Politiken, bei denen das geschieht, wollen wir Prioritätspolitiken nennen. Ähnlich wie bei den LIFO-Politiken können die Prioritäten abgemildert sein. Man kann einem Auftrag, der nicht von der bevorzugten Bedienungsstation kommt, der aber allein zur Bedienung ansteht, eine Zeitvorgabe von k Intervallen gewähren. Der bevorzugte Auftrag kann dann erst die Regie erhalten, wenn der benachteiligte Auftrag erledigt ist oder die Zeitvorgabe abgelaufen ist. Der bevorzugte Auftrag erhält dann natürlich die maximal zulässige Quantenzahl. Die Prioritätspolitiken bezeichnen wir mit PRIO(1,k) bzw. PRIO(2,k), wenn sie Aufträge von B_1 bzw. B_2 bevorzugen und dem benachteiligten k Quanten Vorgabe gewähren.

Schließlich gibt es noch eine Klasse von Politiken, die wir als Time-Sharing-Politiken bezeichnen wollen. Diese bedienen abwechselnd einen Auftrag von B_1 in l Zeitintervallen und einen von B_2 in m Zeitintervallen, solange beide Aufträge zur Bearbeitung anstehen. Wir können noch vorschreiben, daß ein Auftrag von B_1 (B_2), der allein im System ist, j (k) Zeitvorgabe erhält. Eine derartige Politik werde mit TS(1,j,k,l,m) bzw. TS(2,j,k,l,m) bezeichnet, falls der Time-Sharing-Zyklus mit dem Auftrag von B_1 bzw. B_2 beginnt.
Die folgende Tabelle beschreibt die angegebenen "intuitiven" Politiken mit Hilfe der oben eingeführten Entscheidungsvariablen.

	FIFO (1)	FIFO (2)	LIFO (k,l,1)	LIFO (k,l,2)	PRIO (1,k)	PRIO (2,k)	TS(1,j, k,l,m)	TS(2,j k,l,m)
d(1,1)	M	M	k	k	M	k	j	j
d(1,2)	M	M	l	l	k	M	k	k
d(1,3)	-M	M	-M	M	-M	M	-l	m
d(2,1)	M	M	l	l	k	M	k	k
d(2,2)	M	M	-M	-M	-M	M	-l	m
d(3,1)	M	M	k	k	M	k	j	j
d(3,2)	-M	-M	M	M	-M	M	-l	m
d(4,1)	-M	M	-M	M	-M	M	-l	m
d(5,1)	M	M	k	k	M	k	j	j
d(5,2)	-M	-M	M	M	-M	M	m	-l
d(5,3)	M	M	l	l	k	M	k	k
d(6,1)	-M	-M	-M	-M	-M	M	m	m
d(6,2)	M	M	l	l	k	M	k	k
d(7,1)	M	M	l	l	k	M	k	k
d(7,2)	M	M	-M	-M	-M	M	-l	m
d(7,3)	M	M	k	k	M	k	j	j
d(8,1)	M	M	M	M	-M	M	-l	-l
d(8,2)	M	M	k	k	M	k	j	j
d(4k'+1,1)	M	M	l	l	k	M	k	k
d(4k'+2,1)	M	M	l	l	k	M	k	k
d(4k'+3,1)	M	M	k	k	M	k	j	j
d(4k'+4,1)	M	M	k	k	M	k	j	j

In einfachen Spezialfällen kann man aus der Gleichung (17) für den Gewinn eine optimale Politik sofort ablesen. Wir geben einige Beispiele an:

a) Sei KN $\neq$ 0, KA1=KA2=KU1=KU2=KW1=KW2 = 0, WMAX = 1. Es verursacht also nur die Verarbeitung des Weckersignals Kosten und es darf maximal ein Zeitquantum zugeteilt werden. Dann gilt $G = - KN \sum_{i=1}^{8} \tilde{\pi}_i = - KN$, denn bei WMAX = 1 sind nur 8 Zustände möglich. Der Gewinn ist in diesem Fall also unabhängig von der benutzten Politik, d.h. jede Politik ist optimal.

b) Ist KU1 > 0 und sind die übrigen Kosten gleich 0, so ist der Gewinn

$$- G = [\tilde{\pi}_5 P_2 (1-Q_1) \sum_{l=1}^{M} d_l(5,2) + \tilde{\pi}_6 (1-Q_1) \sum_{l=1}^{M} d_l(6,1)] KU1.$$

Der Ausdruck ist größer oder gleich 0. Der Wert 0 wird angenommen, falls $d(5,2) < 0$ und $d(6,1) < 0$ sind. Die Tabelle der intuitiven Politiken zeigt sofort, daß die Politiken FIFO(1), FIFO(2) und PRIO(1,k) diese Bedingung erfüllen und daher optimal sind.

c) Ganz analog erkennt man die Optimalität von FIFO(1),FIFO(2) und PRIO(2,k) für den Fall, daß nur die Unterbrechung eines Auftrags von B_2 Kosten verursacht.

d) Sind beide Unterbrechungskosten positiv, die übrigen Kosten jedoch 0, so kommt man auf demselben Weg wie in b) zum Nachweis der Optimalität von FIFO(1) und FIFO(2). Das kann man aber auch mit Hilfe von (18) und (19) aus b) und c) schließen. Intuitiv sind die erhaltenen Resultate ebenfalls sofort einsichtig. Da nur die Unterbrechungen Kosten verursachen, braucht man nur Unterbrechungen vollständig zu vermeiden, wie es die

angegebenen optimalen Politiken tun.

Im allgemeinen ist es jedoch nicht so einfach, eine optimale Politik zu finden. Es wurde daher ein ALGOL-Programm geschrieben, das nach dem Howardschen Algorithmus für den nichtdiskontierten Fall für gegebene Wahrscheinlichkeits- und Kostenparameter den optimalen Gewinn und optimale Politiken für das Zwei-Benutzer-Modell bestimmt. Wir geben einige Resultate an, die mit diesem Programm erhalten wurden.

In Figur 3 sind die optimalen Kosten als Funktion der Wartezeitkosten KW2 für verschiedene WMAX-Werte aufgetragen. Die mittlere Rechenzeit und die mittlere Denkzeit an beiden Bedienungsstationen ist gleich 10 Zeiteinheiten. Die Verarbeitung des Weckersignals kostet eine Kosteneinheit, ebenso viel jedes Intervall der Anwesenheit eines beliebigen Auftrags und das Warten eines Auftrags von B_1.

Für jeden der betrachteten WMAX-Werte ist die Kostenkurve aus Geradenstücken zusammengesetzt. Das liegt daran, daß jeweils für einen ganzen KW2-Bereich dieselbe Politik optimal ist. Aus unseren theoretischen Überlegungen wissen wir aber, daß der Gewinn (d.h. die negativen Kosten) dieser Politik linear in KW2 ist. Das mußten die Rechnungen natürlich bestätigen. Dagegen zeigt uns Figur 4, daß die Abhängigkeit der Kosten einer Politik (dort PRIO(2,5) bei WMAX=5) von den Wahrscheinlichkeitsparametern keineswegs linear ist. Aber auch das haben wir nach der Theorie nicht anders erwartet. In den in Figur 3 betrachteten Fällen ergaben sich als optimale Politiken immer Prioritätspolitiken.

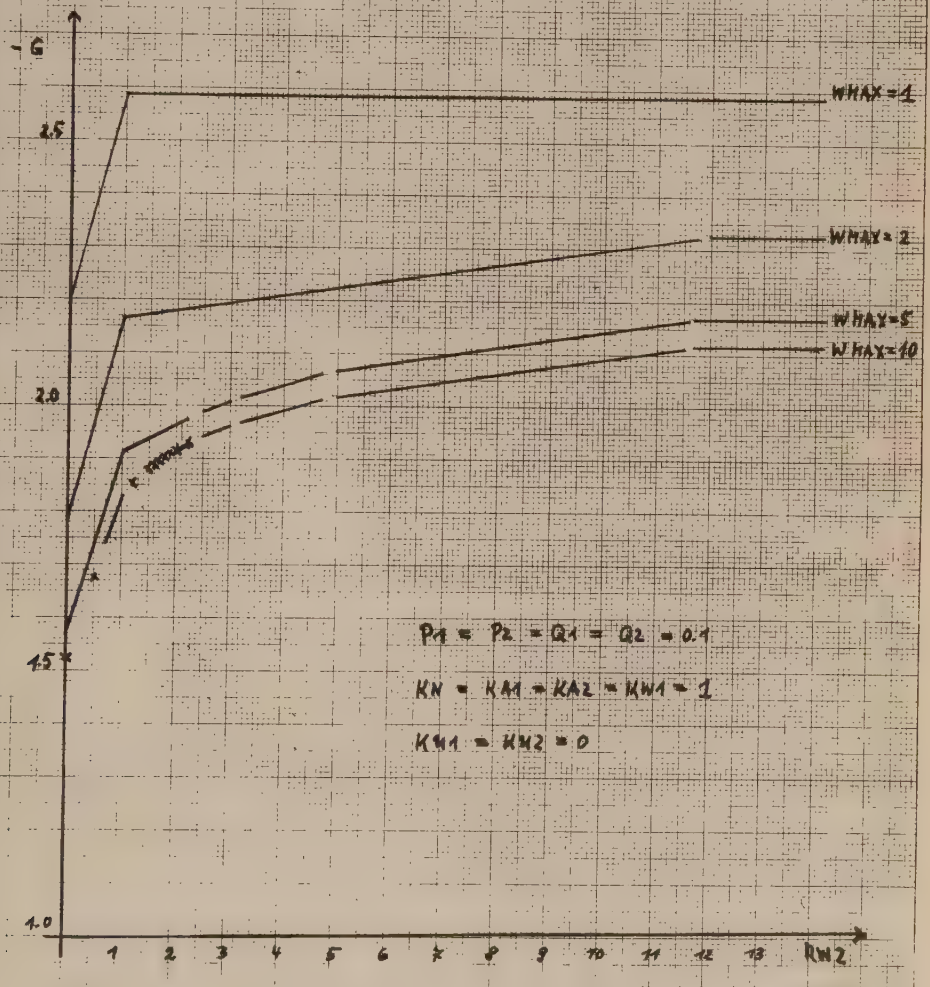

Fig. 3 Der optimale Gewinn als Funktion von KW2 und WMAX

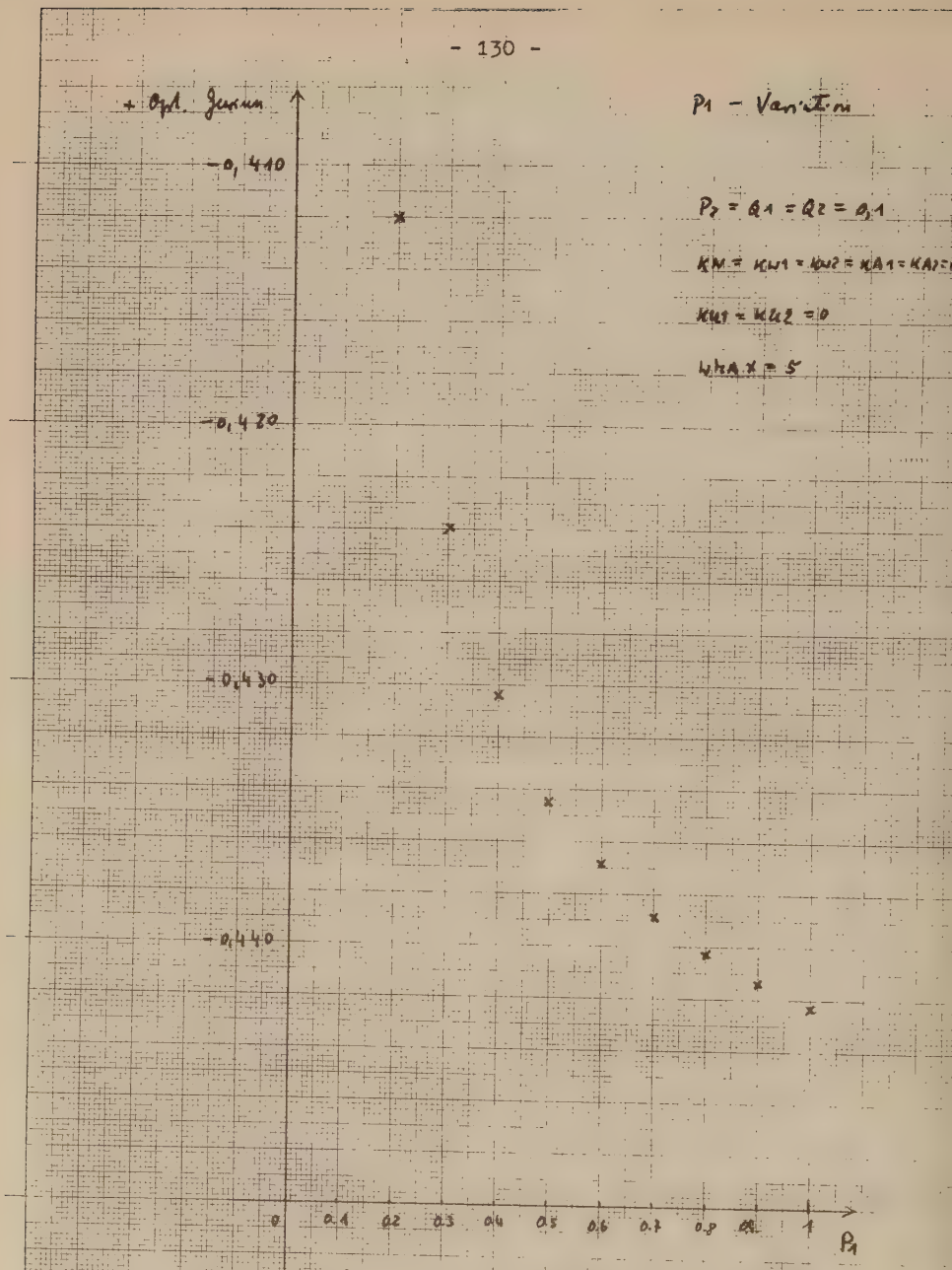

Fig. 4 Der Gewinn der Politik PRIO(2,5) als Funktion von P1

Bei WMAX = 1 ist PRIO(1,1) für KW2 $\leq$ 1 und sonst PRIO(2,1) optimal. Weitere Beispiele enthält die folgende Tabelle.

WMAX	KW2	Optimale Politik
2	0 - 1	PRIO(1,2)
2	1 - 11.7	PRIO(2,2)
2	11.8 - ∞	PRIO(2,1)
5	0 - 1	PRIO(1,5)
5	1 - 2.2	PRIO(2,5)
5	2.2 - 3	PRIO(2,4)
5	3.2 - 4.7	PRIO(2,3)
5	5 - 11.7	PRIO(2,2)
5	11.8 - ∞	PRIO(2,1)
10	0	PRIO(1,5)
10	0.5	PRIO(1,9)
10	0.7 - 1	PRIO(1,10)
10	1 - 1.2	PRIO(2,10)
10	1.5	PRIO(2,9)
10	1.6	PRIO(2,8)
10	1.7	PRIO(2,7)
10	1.8 - 1.9	PRIO(2,6)
10	2 - 2.2	PRIO(2,5)
10	2.5 - 3	PRIO(2,4)
10	3.2 - 4.7	PRIO(2,3)
10	5 - 11.6	PRIO(2,2)
10	11.8 - ∞	PRIO(2,1)

Die Intervallgrenzen, bei denen sich die optimalen Politiken ändern, lassen sich natürlich mit etwas zusätzlichem Rechenaufwand genauer bestimmen. Für unsere mehr qualitativen Überlegungen ist ihre genaue Lage aber unwichtig. Der Praktiker wird bemerken, daß auch von seinem Standpunkt eine sehr präzise Kenntnis der Umschlagstellen nicht erforderlich ist,

da die Kostenfunktionen der Politiken in weiten Bereichen ziemlich flach verlaufen.

Wir bemerken, daß die Kostenfunktion dort, wo die Politik PRIO(2,1) optimal ist, parallel zur KW2-Achse verläuft. Intuitiv ist dieses Resultat klar: Bei großen Wartezeitkosten ist es günstig, dem betreffenden Benutzer höchste Priorität zu geben, wie es PRIO(2,1) tut. Dann muß dieser Benutzer überhaupt nicht warten, die Kosten KW2 fallen also nie an. Bei kleineren KW2-Werten stehen diesem Verfahren aber die Kosten zur Verarbeitung des Weckersignals entgegen. Der nicht bevorzugte Benutzer erhält bei PRIO(2,1) nämlich jeweils nur ein Zeitquantum, nach dessen Ablauf bereits wieder die Weckersignalkosten anfallen. Bei fallenden KW2-Werten muß daher der benachteiligte Benutzer eine immer größere Zeitvorgabe erhalten. Bei KW2 < 1 sind dann die Wartekosten für den Benutzer von B_1 (KW1 = 1) größer, so daß eine Bevorzugung dieses Benutzers erfolgt.

Eine weitere in Fig. 3 dargestellte Erscheinung ist theoretisch klar. Wenn wir WMAX, die maximal zulässige Zeitvorgabe, vergrößern, so vergrößert sich die Menge der zulässigen Politiken. Von den optimalen Kosten wird man daher erwarten, daß sie geringer werden. Die Abnahme ist in Fig. 3 deutlich zu sehen.

Bei diesen numerischen Berechnungen haben wir stets eine optimale Politik unter den beschriebenen "intuitiven" Politiken finden können. Wir vermuten, daß das für unser Zwei-Benutzer-Modell allgemein gilt. Ein analytischer Beweis konn-

te aber leider nicht gefunden werden.

Bei keinem Beispiel mit positiven Kosten war eine Time-Sharing-Politik zur Erreichung des Optimums nötig. In der Praxis ist man demgegenüber aber überzeugt, daß Time-Sharing Vorteile bringt. Wir müssen daher annehmen, daß unser Modell einige wesentliche praktische Gegebenheiten außer acht läßt. Intuitiv ist dieser Mangel leicht einzusehen. Stehen in unserem Modell Aufträge von beiden Bedienungsstationen zur Bearbeitung an, so befinden sie sich in einer bestimmten Konkurrenzsituation, in der aufgrund der Bewertung ein Auftrag als der wichtigere erscheint. Ist dieser nun einige Intervalle bedient worden, ohne daß er beendet worden ist, so ist bei der nächsten Rechnerkernvergabe die Konkurrenzsituation überhaupt nicht verändert; die Wahrscheinlichkeiten für die Beendigung und die Wartekosten sind genau dieselben wie vorher. Es ist also kein Grund zu sehen, der es erforderlich machen sollte, einen Auftrag zugunsten des anderen zu unterbrechen. In einem Teilnehmersystem ist die Situation aber völlig verschieden. Dort spielt es für die Konkurrenzsituation sehr wohl eine Rolle, wie lange ein Auftrag bereits wartet. Das in 5.1 und 5.3 beschriebene Modell gestattet es, diese zeitlichen Veränderungen zu erfassen.

In ganz rudimentärer Form können wir die Zeitabhängigkeit der Kosten aber auch in unserem einfachen Zwei-Benutzer-Modell erreichen. Die Kosten KW1 (KW2) entstehen in einem Zeitintervall, in dem ein Auftrag von B_1 (B_2) ansteht, aber nicht bedient wird. Die Unterbrechungskosten KU1 (KU2) entstehen in dem Intervall nach der Unterbrechung eines Auftrags von

B_1 (B_2). Die Gesamtkosten betragen im ersten Intervall nach einer Unterbrechung also KW1 + KU1 (KW2 + KU2), in den übrigen nur KW1 bzw. KW2. Wählen wir nun KU1 < 0 und KU2 < 0, so können wir diese Wahl so interpretieren, daß das erste Wartezeitintervall nach einer Unterbrechung weniger schlimm ist als die weiteren.

Die folgende Tabelle, die wieder mit dem oben erwähnten Programm erhalten wurde, zeigt, daß sich nun auch Time-Sharing-Politiken als optimal erweisen. Gemeinsam sind allen aufgeführten Beispielen die folgenden Parameterwerte:

WMAX = 5, P1 = P2 = Q1 = Q2 = 0.1, KN = 0.1, KA1 = KA2 = 0.

KU1	KU2	KW1	KW2	Opt.Gewinn	Opt. Politik
-0.9	-0.9	0.9	0.1	0.0617	TS(1,2,1,1,1)
-0.9	-1.2	0.9	0.5	0.0379	TS(1,2,1,1,1)
-0.9	-1.2	0.9	0.1	1.1855	TS(1,2,1,1,1)
-0.9	-0.9	0.9	0.5	-0.0190	TS(1,2,2,1,1)
-0.9	-0.5	0.9	0.5	-0.0925	TS(1,2,2,1,1)
-0.9	-0.9	0.9	0.9	-0.0974	TS(1,2,2,1,1)
-0.9	-0.5	0.9	0.9	-0.1710	TS(1,2,2,1,1)
-0.8	-0.9	0.9	0.9	-0.1158	TS(2,2,2,1,1)
-0.9	-0.1	0.9	0.9	-0.2446	TS(1,2,2,1,1)
-0.9	-0.7	0.9	0.1	0.0243	TS(1,3,1,1,1)
-0.9	-0.5	0.9	0.1	-0.0126	TS(1,3,2,1,1)
-0.9	-0.3	0.9	0.5	-0.1290	TS(1,3,2,1,1)
-0.9	-0.1	0.9	0.5	-0.1649	TS(1,3,2,1,1)
-0.9	0.1	0.9	0.9	-0.2805	TS(1,3,3,1,1)
-0.9	0.1	0.9	0.5	-0.2003	TS(1,4,2,1,1)
-0.9	0.5	0.9	0.9	-0.3479	TS(1,4,5,1,1)
-0.9	-0.3	0.9	0.1	-0.0477	TS(1,5,2,1,1)
-0.9	-0.1	0.9	0.1	-0.0819	TS(1,5,2,1,1)
-0.9	0.2	0.9	0.5	-0.2176	TS(1,5,2,1,1)
-0.9	0.5	0.9	0.5	-0.2550	TS(1,5,3,5,1)

Außer bei der letzten der angegebenen Politiken werden die Benutzeraufträge abwechselnd je eine Zeiteinheit bedient, solange beide noch Service verlangen. Man kann diese Politiken also im wesentlichen als einfache Round-Robin-Politiken verstehen. Was sie von den bekannten Round-Robin-Politiken unterscheidet, ist die differenzierte Behandlung eines einzeln anstehenden Auftrags. Je nach der vorliegenden Bewertung und nach seiner Herkunft kann er ganz verschieden behandelt werden. Die zuletzt aufgeführte Politik weist darüber hinaus den Aufträgen auch beim Time-Sharing jeweils verschiedene Zeitquanten zu.
Die Erkenntnisse aus den numerischen Rechnungen können wir wie folgt zusammenfassen:

1. die Gestalt der optimalen Politik hängt von den vorliegenden Wahrscheinlichkeits- und Kostenparametern ab;
2. bei Kostenfunktionen, die sich mit der Abarbeitung von Aufträgen ändern, erweisen sich häufig Time-Sharing-Politiken als optimal;
3. selbst bei unserem einfachen Zwei-Benutzer-Modell erfahren die Benutzeraufträge im System eine differenziertere Behandlung als bei den üblicherweise in der Literatur betrachteten Zuteilungsalgorithmen.

Die Ausführungen dieses letzten Kapitels haben gezeigt, wie man Optimierungsprobleme bei Teilnehmerrechensystemen als spezielle stochastische Entscheidungsmodelle formulieren und in einfachen Fällen numerisch lösen kann. Auch n-Benutzer-Modelle sind im Prinzip auf diese Weise zu behandeln. Die

praktische Brauchbarkeit dieses Verfahrens wird jedoch durch die großen Speicherplatzforderungen der entsprechenden Programme erheblich behindert. Einen ersten Eindruck dazu vermitteln die folgenden Zahlen. Für ein n-Benutzermodell, das analog zu unserem Beispiel (5.4) organisiert ist, liegt die Speicherplatzforderung ganz grob in der Größenordnung von $n^2 \cdot 2^{2n-2} \cdot WMAX^2$ Worten. Für ein Zwei-Benutzer-Modell nach (5.3), bei dem die dort aufgeführten Maximalwerte alle gleich WMAX sind, benötigt ein (existierendes) Programm ganz grob $WMAX^{12}$ Worte. Zur Überwindung dieser Schwierigkeiten und auch zum Verständnis der Struktur der optimalen Politiken sind noch weitere Überlegungen nötig.

6. LITERATURVERZEICHNIS

ADIRI,I. (1969): Computer time-sharing queues with priorities. JACM 16, 631-645

ADIRI,I. und AVI-ITZHAK,B. (1969): A time-sharing queue with a finite number of customers. JACM 16, 315-323.

AHO,A.V.; DENNING,P.J. und ULLMANN,J.D. (1971): Principles of optimal page replacement. JACM 18, 80-93.

ARDEN,B.W.; GALLER,B.A.; O'BRIEN,T.C. und WESTERVELT,F.H. (1966): Program and adressing structure in a time-sharing environment. JACM 13, 1-16.

BELADY,L.A. (1966): A study of replacement algorithms for a virtual storage computer.IBM Syst. J. 5, 78-101

BELLMAN,R. (1957): Dynamic Programming. Princeton.

BELLMAN,R. und S.E.DREYFUS (1962): Applied Dynamic Programming.Princeton.

BILLING,H. (1968): Grundgedanken und Probleme des Teilnehmer-Rechensystems. In: Teilnehmerrechensysteme, Beihefte zur Zeitschrift "Elektr.Rechenanlagen", Bd. 17, 11-28

BLACKWELL,D. (1965): Discounted Dynamic Programming. Ann. Math. Statist. 36, 226-235

BLAKE,K. und GORDON,G. (1964): Systems simulation with digital computers. IBM Syst. J. 3, 14-20

BRAWN,B. und GUSTAVSON,F. (1968): Program behavior in a paging envoronment, Proc.AFIPS 1968 FJCC,1019-1032.

BRYAN,G.E. (1967): JOSS, 20 000 hours at the console- a statistical summary. Proc. AFIPS FJCC, 769-777

CHANG,W. (1966): A queueing model for a simple case of time-sharing. IBM Syst. J. 5, 115-125.

CHANG,W. (1968): Queues with feedback for time-sharing computer system analysis. Oper. Res. 16, 613-627.

CHARZAN,D.; KONHEIM,A. und WEISS,B. (1968): A note on time-sharing. J. Comb. Theory 5, 344-369.

CIUCU,G. und THEODORESCU,R. (1960): Processes with complete connections. Bucharest.

COFFMAN,E.G. (1967): Studying multiprogramming systems with the queueing theory. Datamation 13, 47-54.

COFFMAN,E.G. (1968): Analysis of time-sharing algorithms designed for limited swapping. JACM 15, 341-353.

COFFMAN,E.G. und KLEINROCK,L. (1968): Feedback queueing models for time-shared systems. JACM 15, 549-576.

COFFMAN,E.G. und KLEINROCK,L. (1968): Computer scheduling methods and their countermeasures. Proc. AFIPS SJCC, 11-21.

COFFMAN,E.G. und VARIAN,L.C. (1968): Further experimental data on the behavior of programs in a paging environment. Comm. ACM 11, 471-474.

COFFMAN,E.G. und WOOD,R.C. (1966): Interarrival statistics for time-sharing systems. Comm. ACM 9, 500-503.

CORBATÒ,F.J. (1963): The compatible time-sharing system: a programmer's guide. MIT-Press, Cambridge, Mass.

COX,D.R. und SMITH,W.L. (1961): Queues. London, New York.

CRISMAN,P.A. (1965): The compatible time-sharing system: a programmer's guide. 2nd Ed., MIT-Press, Cambridge, Mass.

DE CANI,J.S. (1964): A dynamic programming algorithm for embedded Markov chains when the planning horizon is at infinity. Man. Sci. 10, 716-733.

DENNING,P.J. (1968): The working set model for program behavior. Comm. ACM 11, 323-333.

DENNING,P.J. (1968): A statistical model for console behavior in multiuser computers. Comm. ACM 11, 605-612.

DENNIS,J.B. (1965): Segmentation and the design of multi-programmed computer systems. JACM 12, 589-602.

DERMAN,C. (1962): On sequential decisions and Markov chains. Manag. Sci. 9, 16-24.

DERMAN,C. (1966): Denumerable state Markovian decision processes - average cost criterion. Ann.Math.Statist. 37, 1545-1553.

DOOB,J.L. (1953): Stochastic processes. New York, London.

DORN,W.S. (1969): Editor's preview. Comput. Surv. 1, 81-84.

DUBNER,H. und J.ABATE (1970):TICKETRON - A successful operating system without an operating system. Proc. AFIPS SJCC, 143-155.

DYNKIN,E.B. (1965): Controlled random sequences. Theor. Prob. 10, 1-14.

ERLANG,A.K. (1918): Lösung einiger Probleme der Wahrscheinlichkeitsrechnung von Bedeutung für die selbsttätigen Fernsprechämter. Elektrotechn.Z. 504-508

FANO,R.M.und CORBATÒ,F.J. (1966): Time-sharing on computers. Scientific American 215,3 128-140.

FINE,G.H.; C.W.JACKSON und P.V.McISAAC (1966): Dynamic program behavior under paging. Proc. 1966 ACM Nat.Conf.

FINE,G.H. und P.V.McISAAC (1966): Simulation of a time-sharing system. Manag. Sci. 12, B 180-194.

FLORES,I. (1969): Computer organization. Englewood Cliffs, N.J.

FOTHERINGHAM,J. (1961): Dynamic storage allocation in the ATLAS computer including an automatic use of backing store. Comm. ACM 4, 435-436.

GAVER,D.P. (1967): Probability models for multiprogramming computer systems. JACM 14, 423-438.

GNEDENKO,B.V. und KOVALENKO,I.N. (1966): Introduction to queueing theory. Moskau.

GOLD,M.M. (1969): Time-sharing and batch-processing: An experimental comparison of their values in a problem-solving situation. Comm.ACM 12, 249-259.

HABERMANN,A.N. (1969): Prevention of system deadlocks. Comm. ACM 12, 373-377, 385.

HASSITT,A. (1967): Computer programming and computer systems. New York. London.

HAVENDER,J.W. (1968): Avoiding deadlocks in multitasking systems. IBM Syst. J. 7, 74-84.

HELLERMAN,H. (1969): Some principles of time-sharing scheduler strategies. IBM Syst. J. 8, 94-117.

HINDERER,K. (1967): Zur Theorie stochastischer Entscheidungsmodelle. Habilitationsschrift, Univ. Stuttgart.

HINDERER,K. (1970): Foundations of non-stationary dynamic programming with discrete time parameter. Berlin, Heidelberg, New York.

HOWARD,R.A. (1960): Dynamic programming and Markov processes. New York.

IONESCU TULCEA,C.T. (1949): Mesures dans les espaces produits. Atti Acad. Nat. Lincei Rend. 7, 208-211.

IOSIFESCU,M. (1963): Random events with complete connections with an arbitrary set of states. Comunarile Acad.R.P.R. 13, 579-581.

IOSIFESCU,M. und THEODORESCU,R. (1969): Random processes and learning. Berlin, Heidelberg, New York.

JESSEN,E. (1968): Das Betriebssystem des Rechners TR 440. In: Teilnehmer-Rechensysteme. Beihefte zur Zeitschrift "Elektr. Rechenanlagen", Bd.17, 115-123.

JEWELL,W.S. (1963): Markov-renewal Programming, I und II. Oper. Res. 11, 938-948, 949-971.

KARLIN,S. (1953): Some random walks arising in learning models. Pacific J. Math. 3, 725-756.

KILBURN,T.; PAYNE,R.B. und HOWARTH,D.J. (1962): The ATLAS supervisor. Computers- key to total systems control. 279-294.

KLEINROCK,L. (1967): Time-shared systems: a theoretical treatment. JACM 14, 242-261.

KLEINROCK,L. (1968): Certain analytical results for time-shared processors. Proc. 1968 IFIP Congr.,838-845.

KLEINROCK,L. (1970): A continuum of time-sharing scheduling algorithms. AFIPS SJCC, 453-458.

KLEINROCK,L. (1970): Analytic and simulation methods in computer network design. Proc. AFIPS SJCC.

KLEINROCK,L. und E.G.COFFMAN (1967): Distribution of attained service in time-shared systems. J. Comput. Syst. Sci. 1, 287-298.

KONHEIM,A.G. (1968): A note on time sharing with preferred customers. Z. Wth. verw. Geb. 9, 112-130.

KRISHNAMOORTHI,B. und R.C.WOOD (1966): Time-shared computer operations with both interarrival and service times exponential. JACM 13, 317-338.

KÜMMERLE,K. (1970): Simulation of the performance of computer systems. Elektr. Rechenanlagen 12, 324-328.

KÜSPERT,H.J. und G.MARTE (1970): Optimale Rechenzeit-Zuteilung bei einem Teilnehmer-Rechensystem mit jeweils einer Aufgabe im Arbeitsspeicher. Elektr. Rechenanl. 12, 155-162.

LEHMANN,F. (1966): Zur Optimierung von Time-Sharing Systemen. Bericht an die Deutsche Forschungsgemeinschaft. München.

LEHMANN,F. (1967): Allgemeiner Bericht über Monte-Carlo-Methoden. Blätter d.Dt.Ges.f.Versmath. VIII,431-456.

LEHMANN,F. (1968): Zur Optimierung von Time-Sharing-Systemen. In: Teilnehmer-Rechensysteme, Beihefte zur Zeitschrift "Elektr. Rechenanlagen", Bd. 17, 255-265.

MANNE,A.S. (1960): Linear programming and sequential decisions. Manag. Sci. 6, 259-267.

MARTE,G. (1970): Optimal time scheduling for time-shared computer systems with piecewise exponential computing time distribution. Intern.Comp.Symp.ACM, Bonn.

McCARTHY,J.; BOILEN,S.; FREDKIN,E. und LICKLIDER,J.C.R. (1963): A time-sharing debugging system for a small computer. Proc. AFIPS SJCC, 51-57.

McKINNEY,J.M. (1969): A survey of analytical time-sharing models. Computing Surveys 1, 105-116.

MILLER,B.L. und VEINOTT,A.F. (1969): Discrete dynamic programming with a small interest rate. Ann. Math. Statist. 40, 366 - 370.

NEVEU,J. (1969): Mathematische Grundlagen der Wahrscheinlichkeitstheorie. München, Wien.

NICKEL,K. (1968): Das Vielfach-Konsol-System HYDRA X8 Karlsruhe. In: Teilnehmer-Rechensysteme. Beihefte zur Zeitschrift "Elektr. Rechenanlagen", Bd. 17, 125-133.

NIELSEN,N.R. (1967): The simulation of time sharing systems. Comm. ACM 10, 397-412.

NOLLE,F.N. (1970): Datenfernverarbeitung. Köln-Braunsfeld.

OLIVIER,G. (1967): Optimale Zeitzuteilung für wartende Rechenaufgabe. Elektr. Rechenanl. 9, 218-224.

ONICESCU,O. und MIHOC,G. (1935): Sur les chaines statistiques. C.r.Acad.Sci.Paris 200, 511-512.

ONICESCU,O. und MIHOC,G. (1935): Sur les chaines de variables statistiques. Bull.Sci.Math. 59,174-192

ONICESCU,O. und MIHOC,G. (1936): Sur les chaines statistiques. C.r.Acad.Sci.Paris 202, 2031-2033.

PAUL,M. und URICH.W. (1966): Vielfachzugriff-Systeme. Die Naturwissenschaften 53, 325-329.

PENNY,J.P. (1966): An analysis, both theoretical and by simulation, of a time-shared computer system. Computer J. 9, 53-59.

RANDELL,B. (1969): A note on storage fragmentation and program segmentation. Comm. ACM 12, 365-369,372.

RANDELL,B. und KUEHNER,C.J. (1968): Dynamic storage allocation systems. Comm. ACM 11, 297-306.

RASCH,P.J. (1970): A queueing theory study of round-robin scheduling of time-shared computer systems. JACM 17, 131-145.

RIORDAN,J. (1962): Stochastic service systems. New York, London.

ROSENBERG,A.M. (1964): Computer-usage accounting for generalised time-sharing systems. Comm. ACM 7, 304-308.

SACKMAN,H.; W.J.ERIKSON und E.E. GRANT (1968): Exploratory experimental studies comparing online and offline programming performance. Comm. ACM 11, 3-11.

SCHATZOFF,M.; TSAO,R. und WIIG,R. (1967): An experimental comparison of time-sharing and batch processing. Comm. ACM 10, 261-265.

SCHECHER,H. (1970): Struktureller Aufbau digitaler Rechenanlagen. Skriptum, Teil III, TU München.

SCHERR,A.L. (1966): Time-sharing measurement. Datamation, 22-26.

SCHERR,A.L. (1967): An analysis of time-shared computer systems. Res.Mon.No. 36, MIT Press,Cambridge,Mass.

SEAMAN,P.H. und SOUCY,R.C. (1969): Simulating operating systems. IBM Syst. J. 8, 264-279.

SELWYN,L.L. (1970): Computer resource accounting in a time sharing environment. Proc. AFIPS 1970 SJCC, 119-130.

SHEMER,J.E. (1967): Some mathematical considerations of time-sharing scheduling algorithms. JACM 14, 262-272.

SIRJAEV,A.N. (1967): Some new results in the theory of controlled random processes. Trans. 4th Prague Conf. (Russisch; englische Übersetzung in: Selected Translations in Mathematical Statistics and Probability, 8, 1970, 49-130).

SPIES,P.P. (1967): Teilnehmer-Rechensysteme. Elektr.Rechenanl. 9, 117-122.

SPIES,P.P. (1970): Analyse eines Warteschlangenmodells für Teilnehmer-Rechensysteme. Elektr.Rechenanl. 12, 306-314.

STRAUCH,R.E. (1966): Negative dynamic programming. Ann. Math. Statist. 37, 871-890.

TAKACS,L. (1962): Introduction to the theory of queues. New York.

TELEFUNKEN Teilnehmer-Betriebssystem BS 3, 1970.

THEODORESCU,R. (1960): Nouvelle interprétation de certains modèles stochastiques pour apprendre. Rend.Accad.Naz.dei Lincei, Cl.Sci.fis.mat.nat., Ser. 8, 28, 153-155.

VARIAN,L.C. und COFFMAN,E. (1967): An empirical study of the behavior of programs in a paging environment. Preprints ACM Symp. Operating Syst. Principles, Gatlinburg, Tennessee.

WILKES,M.V. (1970): Time-Sharing-Betrieb bei digitalen Rechenanlagen. München.

LEBENSLAUF

Am 29. Juli 1938 wurde ich als Sohn des Steuerrats Fritz Lehmann und seiner Ehefrau Helene, geborene Walczak, in Ahaus (Westf.) geboren. Im Herbst 1944 trat ich dort in die Volksschule ein.

Von 1949 bis zum Sommer 1954 besuchte ich das Alexander-Hegius-Gymnasium in Ahaus, dann das Staatliche Gymnasium Paulinum in Münster (Westf.), wo ich am 26.2.1958 die Reifeprüfung bestand.

Im Sommersemester 1958 und im Wintersemester 58/59 studierte ich Mathematik und Physik an der Westfälischen Wilhelms-Universität in Münster. Ich hörte dort Vorlesungen bei den Professoren Kappler, Peterson, Remmert, Sommer und anderen. Vom Sommersemester 1959 an setzte ich mein Studium an der Ludwig-Maximilians-Universität in München fort. Meine Lehrer waren dort unter anderen die Professoren Aumann, Bopp, Fässler, Heinz, Koecher, Richter, Schlüter, Seebach und Stein. Am 18. Dezember 1964 legte ich die Diplom-Hauptprüfung im Fach Mathematik ab.

Vom 1.1.1965 bis zum 30.11.1967 gehörte ich als wissenschaftlicher Mitarbeiter dem Leibniz-Rechenzentrum der Bayerischen Akademie der Wissenschaften an. Vom 1.12.1967 bis zum 30.4.1970 war ich Verwalter einer wissenschaftlichen Assistentenstelle am Lehrstuhl für Mathematische Statistik an der Universität München. Am 1.5.1970 kehrte ich zum Leibniz-Rechenzentrum zurück, wo ich jetzt als wissenschaftlicher Mitarbeiter angestellt bin.

Fritz Lehmann